Settembre 1984 - N. 3 - Lire 3.000 Anno I - N. 3 - Sped. Abb. post. gr. III/70 - CR - Distr. Me Pe

COMMODORE

MENSILE PER UTENTI DI VIC 20 & COMMODORE 64

PRESENTA

systems

COMMODORE

DIRETTORE RESPONSABILE
Agostina Ronchetti

REDATTORE CAPO
Gloriano Rossi

SEGRETARIA DI REDAZIONE
Maura Ceccaroli

GRAFICA e IMPAGINAZIONE
Renato Caruso
Francesco Amatori

FOTO
Franco Vignati

DIFFUSIONE E ABBONAMENTI
Marina

DIREZIONE, REDAZIONE
V.le Famagosta, 75
20142 Milano - Tel. 02/8466675
Autorizzazione del Tribunale di Milano
n. 103 del 25/2/84

STAMPA
Litografica - Busto Arsizio

Concessionario esclusivo
per la diffusione - MEPE spa
Via G. Carcano, 32 - Milano

Spedizione in abbonamento
postale - Gruppo III/70

Prezzo della rivista L. 3.000
Numero arretrato L. 6.000

Abbonamento annuo L. 25.000
I versamenti vanno indirizzati
a: Commodore C.C.
V.le Famagosta, 75 - 20145 Milano,
mediante emissione di assegno bancario utilizzando il c/c postale n.ro 31532203

Per i cambi di indirizzo, indicare, oltre naturalmente al nuovo, anche l'indirizzo precedente, ed allegare alla comunicazione l'importo di L. 500 anche in francobolli.

LA POSTA

• 1) Vorrei sapere come certi programmi, a caricamento avvenuto, girano senza dare il RUN.
2) Da un po' di tempo, il mio floppy 1541, legge i programmi con fatica e molti non li legge. I competenti in materia mi hanno detto che si tratta di allineamento della testina "una cosa da niente" che però mi costerà 3 o 4 mesi di inattività (tale è il tempo di spedizione al più vicino laboratorio), inoltre non è detto che non si sfasino più (anche tra breve tempo). Vorrei sapere se tale taratura, visto la lunghezza della cosa è possibile farla "in casa".
3) Vorrei delle chiarificazioni sull'articolo proteggersi dagli sprotettori in quanto ho perduto una mattinata col provare il SAVE "PROVAAT" senza ottenere nulla o quasi.

(Rizzi Fabrizio - Venezia)

• Esistono materiali su disco per poter regolare l'allineamento della testina del drive. Conoscere qualche sistema.

(Bifolchi Giordano - Montepulciano)

☐ Rispondo ad entrambe le lettere:
1) È possibile tramite la conoscenza del L.M. e quindi di monitor in L.M., procedere al lancio automatico di un programma. Il concetto principe è quello di sfruttare il buffer della tastiera residente nelle locazioni basse della memoria. Se in questo Buffer inserisco il comando RUN e Return abbiamo un avvio automatico. Come ricrearlo?
Ciò è un po' complesso e potrà essere oggetto di un futuro articolo.
2) È ormai risaputo che una parte dei drives 1541 tendono a perdere l'allineamento delle testine, ciò probabilmente è dovuto a qualche componente non propriamente "stagionato".
Cosa fare? Sconsiglio vivamente interventi casalinghi, né tantomeno l'utilizzo di software di dubbia validità.
Un vero ed efficace allineamento di testine viene effettuato con:
1 oscilloscopio a doppia traccia
1 dischetto di allineamento per il drive specifico
1 software per forzare alcune funzioni del 1541
Dopo un riallineamento difficilmente ci si può trovare nuovamente di fronte al medesimo inconveniente.
3) PROVAAT possiede delle T in reverse mode, ciò non corrisponde alla lettera scritta in quel modo, ma al simbolo del comando Delete.

• Durante l'uso del Commodore 64 talvolta ciò che è memorizzato si dissolve nel nulla. Mi accorgo che questo quando tutta la schermata mi si sposta a destra e quando ritorna in posizione normale ho solo le informazioni iniziali del Commodore 64. Ho anche misurato la corrente, in questi casi, ma sembra tutto normale.

(Roberto - Calcinaia)

☐ Probabilmente il problema risiede nel cablaggio del suo Commodore 64 con unità periferiche (disco, registratore ecc. ecc.) infatti se i connettori dei cavi non sono inseriti bene nelle loro prese possono innescare il cosiddetto "System reset" riportando il computer allo stato d'iniziale accensione.
La consigliamo pertanto di rivedere l'integrità dei suoi cavi e se necessario provvedere alla sostituzione degli stessi.

• Vorrei sapere come si può accedere al sistema operativo del C. 64 in modo da inserire altri comandi BASIC personali.

(Luca Prandini - Orbetello)

☐ Implementare il BASIC residente di qualsiasi Personal o Home computer non è così facile ne tantomeno semplice da spiegare.
Con questo non si vuole demoralizzare il caro lettore, ma consiglio prima un approfondimento del linguaggio macchina e del sistema operativo, ciò sarà la chiave di apertura di quella porta per accedere a qualche cosa di più complesso e affascinante.

• Vorrei sapere per cortesia come è possibile conoscere il contenuto delle varie "POKE" del Commodore 64. Vi vorrei chiedere inoltre di essere più semplici nelle vostre spiegazioni, evitando di omettere cose che per voi sono scontate ma che per noi sono talvolta ignote.

(Luca Prandini)

☐ Non è possibile avere un elenco completo delle varie POKE, PEEK o SYS, con i relativi significati. Ciò dipende dal fatto che detti comandi possono avere molteplici aspetti dipendenti dal loro contenuto.
Ciò che è possibile fare e che è in progetto per i prossimi numeri di Commodore è una lunga serie di "spigolature" inerenti a comandi, routinette e notizie varie che per loro natura non giustificano un articolo vero e proprio.

• Chiedo suggerimenti per verificare la fattibilità di una mia idea. Intendersi collegare un piccolo personal (VIC20 od altro) con un contacolpi meccanico, azionato da una levetta, allo scopo di avere una registrazione su disco del numero dei colpi effettuati nell'arco di un certo periodo. Ora chiedo: se è possibile ciò. E se si, a livello hardware o software? Ringrazio anticipatamente.

(Stefano Peruzzo - Verona)

☐ Caro Stefano, una realizzazione di questo genere è sempre possibile. È altrettanto chiaro che deve esistere una certa esperienza sia hardware che software, in quanto la risoluzione del problema vede la necessità di entrambe le parti dell'informatica.
Una soluzione che potrebbe essere ideale vede l'utilizzo di un VIC20 o un Commodore 64 con abbinato con un aggeggino hardware attaccato alla porta Joystick. Ogni volta che si chiuderà il circuito il computer eseguirà le dovute operazioni che avrai previsto. Da queste parole manca solo la pratica che certo non è possibile in così poche righe spiegare.

Pubblichiamo il listato inerente all'articolo "Input controllato" che per un errore di montaggio è apparso capovolto a pag. 40 del n. 2. Ci scusiamo con i lettori.

```
510 PRINT"[HOME]"
520 FOR S=0 TO 19:PRINT:NEXT
530 PRINT TAB(9)"[RVS]FINE INPUT[RV
    OFF]"
540 GET D$:IF D$="" THEN 540
550 PRINT"[CLEAR]"
560 FOR A=1 TO 6
570 PRINT TAB(4)A$(1,A):PRINT
580 NEXT:END
590 :
600 :
610 REM ***************************
620 REM *                         *
630 REM *    INPUT CONTROLLATO    *
640 REM *                         *
650 REM *           DI            *
660 REM *                         *
670 REM *    ERNESTO SIDOTI  E    *
680 REM *                         *
690 REM *      GUIDO MINNECI      *
700 REM *                         *
710 REM ***************************
720 :
730 REM ---------------------------
740 REM  X%..POSIZ. X SUL VIDEO
750 REM  Y%..POSIZ. Y SUL VIDEO
760 REM  L%..LUNGHEZZA DELL' INPUT
770 REM  T%..TIPO (0.ALFA/1.NUM)
775 REM  X$..VARIABILE DI OUTPUT
780 REM ---------------------------
790 :
800 PRINT"[HOME]";:FOR AA=1 TO Y%-1
    :PRINT"[DOWN]";:NEXT:I%=0:IF Y%
    =1 THEN PRINT"[UP]";
810 FOR AA=1 TO X%:PRINT"[RIGHT]";:
    NEXT:FOR AA=1 TO L%:PRINT" ";:N
    EXT:FOR AA=2 TO L%
820 PRINT"[LEFT]";:NEXT:IF T%=1 THE
    N C$=LEFT$(X$,1):IF C$="" OR VA
    L(C$)<>0 THEN C$="+"
830 IF T%=1 THEN X$=C$+RIGHT$(STR$(
    VAL(X$)),LEN(STR$(VAL(X$)))-1):
    I%=1
840 IF T%=1 THEN PRINT"[LEFT]";
850 PRINTX$;:IF T%=1 THEN PRINT"[RI
    GHT]";
860 FOR AA=1 TO LEN(X$):PRINT"[LEFT
    ]";:NEXT
870 GET C$:IF C$<>"" AND FL=0 THEN
    940
880 IF C$="[HOME]" THEN 800
890 K%=PEEK(1024+(Y%-1)*40+X%+I%)
900 IF K%>0 AND K%<127 THEN POKE 10
    24+(Y%-1)*40+X%+I%,K%+128:FL=1
910 IF K%>127 AND K%<255 THEN POKE
    1024+(Y%-1)*40+X%+I%,K%-128:FL=
    0
920 IF C$="" THEN 870
930 IF C$<>"" AND FL=1 THEN 910
940 IF C$="[HOME]" THEN 800
950 C%=ASC(C$):IF C%=145 THEN GOSUB
     1160:GOTO 870
960 IF C%=13 THEN 1100
970 IF T%=1 AND I%=0 THEN IF C%=43
    OR C%=45 THEN 1000
980 IF T%=1 AND (C%=29 OR C%=157) T
    HEN 1000
990 IF T%=1 AND (C%<48 OR C%>57) TH
    EN GOSUB 1160:GOTO 870
000 IF C%=17 THEN GOSUB 1160:GOTO 8
    70
1010 IF C%=157 AND I%<=0 THEN GOSUB
     1160:GOTO 870
1020 IF C%=157 AND I%>0 THEN I%=I%-1
     :PRINT"[LEFT]";:GOTO 870
1030 IF C%=29 AND I%>=L%-1 THEN GOSU
     B 1160:GOTO 870
1040 IF C%=29 AND I%<L% THEN I%=I%+1
     :PRINT"[RIGHT]";:GOTO 870
1050 IF T%=1 AND I%<>1 AND (C%=43 OR
     C%=45) THEN 1080
1060 IF T%=1 AND (C%<48 AND C%>57) T
     HEN GOSUB 1160:GOTO 870
1070 IF C%<32 OR C%>93 THEN GOSUB 11
     60:GOTO 870
1080 IF I%>L%-1 THEN GOSUB 1160:GOTO
      870
1090 I%=I%+1:PRINTC$;:GOTO 870
1100 X$="":IF T%=1 THEN FOR AA=0 TO
     L%-1:C=PEEK(1024+(Y%-1)*40+X%+A
     A):X$=X$+CHR$(C)
1110 IF T%=1 THEN NEXT:GOTO 1150
1120 FOR AA=0 TO L%-1:C=PEEK(1024+(Y
     %-1)*40+X%+AA)
1130 IF  C>27  THEN  C=C-64
1140 X$=X$+CHR$(C+64):NEXT
1150 RETURN
1160 REM  ******* SOUND ********
1170 POKE 54277,8:POKE 54278,17:POKE
      54273,89:POKE 54272,15:POKE 54
     274,0
1180 POKE 54275,9:POKE 54276,65
1190 FOR TT=1 TO 90:NEXT:POKE 54276,
     0:RETURN
```

LO SPACCA BYTE

di **Gloriano Rossi (i2KH)**

Quando ho pensato di fare questo articolo mi sono chiesto se per caso non poteva risultare uno "spaccamento di p...".
Dato l'argomento, ho messo insieme le due cose, ed il titolo che in un secondo momento doveva essere: "computational 3", è invece rimasto, leggermente corretto, come lo vedete in testa alla pagina.
Sui grossi computers o su quegli elaboratori dove si può operare e programmare con linguaggi particolarmente evoluti quale ad esempio il COBOL, è possibile "giocare" con i bytes in una maniera particolare.
Infatti se possiedo un campo numerico è possibile che questo occupi, sia in memoria centrale che in memoria di massa, un numero di bytes inferiore rispetto a quello che si può presupporre.
Quando utilizzo il linguaggio COBOL, posso dichiarare, durante la stesura del programma, che un determinato campo numerico deve essere considerato Computational-3 (in formato abbreviato Comp-3). Alcuni compilatori COBOL prevedono solamente campi numerici con segno, altri invece, un po' più evoluti in valore assoluto.
Sarà il compilatore COBOL e poi in conseguenza il relativo Linker che provvederà a memorizzare opportunamente il numero in oggetto.
La "compattazione" ottenibile in questo linguaggio prevede la riduzione della metà più uno, infatti:

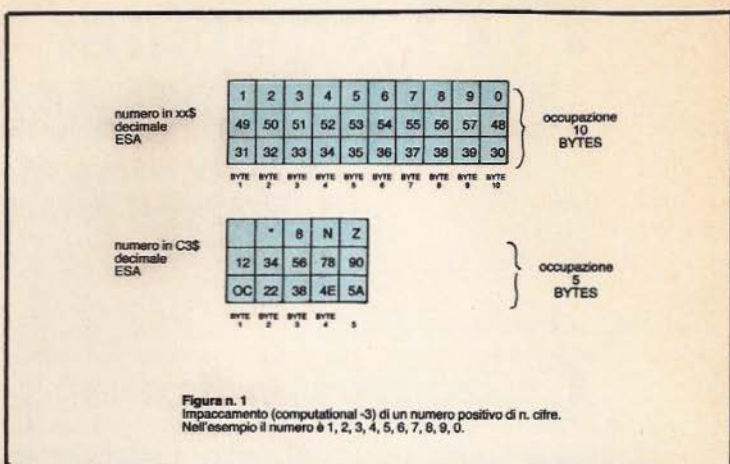
bytes occupati =
lunghezza campo / 2 + 1

Vediamo un esempio:
Consideriamo il numero 25350: se il campo che lo contiene era stato definito a 7 caratteri COMP-3 allora avrò una occupazione di 7/2 + 1 = 4, e se era stato definito senza segno 7/2 = 3.5 e quindi il sistema avrebbe comunque utilizzato 4 bytes.
Ma vediamo in dettaglio cosa può accadere su un computer tipo Commodore:
Se definisco che un campo numerico deve avere una capienza massima tale da poter contenere nove cifre, in condizioni normali tale campo occupa nove bytes di memoria di massa (è bene ricordare che intendiamo la memoria di massa, dischetti o cassette, e non la memoria RAM del computer, in quanto l'organizzazione di quest'ultima relativa a variabili numeriche risulta essere un po' più complessa).
Ora se attribuisco a tale campo la caratteristica di Computational-3, sul file di disco o di nastro, l'occupazione risulterebbe pari a 5 bytes: infatti 9/2 + 1 da proprio 5.5, di questo numero si prende in considerazione esclusivamente la parte intera, quindi 5.
Come è possibile tutto ciò?
Forse siamo tutti a conoscenza, e se uno non lo sa ora lo impara, che il contenuto di un byte è sempre costituito da un numero che va da 0 a 255, tale numero viene scritto in memoria e rappresenta un singolo carattere.
Prendiamo ad esempio un numero qualsiasi: 656667.
Tale numero occupa in memoria esattamente 6 caratteri, bytes.
Ogni byte contiene, in codice ASCII, un numero di codice corrispondente ad una singola cifra; allora il nostro numero sarà scritto in esadecimale in questa maniera: 36 35 36 36 36 37, infatti 36 è uguale a 3 × 16 + 6 = 54, etc.
Il Cinquantaquattresimo codice della tabella ASCII corrisponde appunto al numero 6.
Abbiamo con questo scoperto che qualsiasi carattere, sia esso numerico che alfanumerico, viene sempre associato ad un codice numerico che va da 0 a 255.
Ora se io prendo il numero che abbiamo come esempio e lo divido a gruppi di due, ogni gruppo potrà contenere un numero che potrà andare da 00 a 99. Non possiamo con il medesimo ragionamento prendere in considerazione tre cifre in quanto si avrebbe un numero che potrebbe andare da 000 a 999 e dato che il byte, come già detto, può arrivare ad una cifra massima di 255 si avrebbe una condizione di impossibilità di azione.
Il nostro numero, quindi, sarà scomposto in 65, 66 e 67.
Questi tre nuovi numeri vengono associati al relativo codice ASCII, così:
65 sarà uguale ad A
66 sarà uguale ad B
67 sarà uguale ad C
Allora la stringa "ABC" può corrispondere al numero di partenza 656667, occupando però esattamente la metà dei bytes necessari in origine.
In realtà l'esatta occupazione risultante da tali manovre deve essere calcolata nella seguente maniera:
L = INT((lunghezza originale)/2) + 0.5
Questo è dovuto al fatto che se la lunghezza della stringa di origine corrisponde ad un numero dispari,

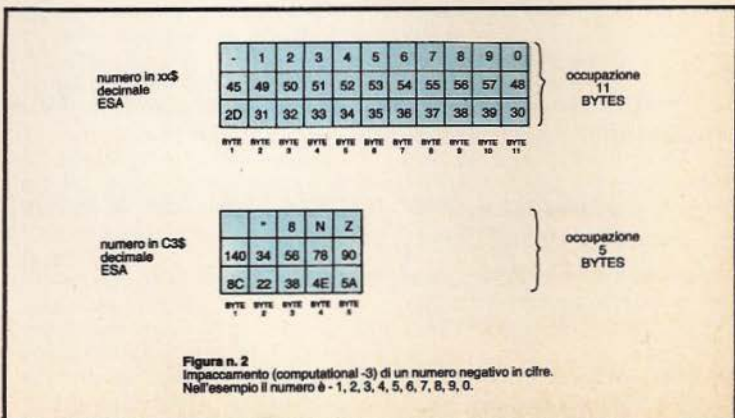
come ad esempio 7, si avrebbe un numero con decimale quale risultato della divisione e quindi... va bene spaccare il byte, ma lasciarlo a metà è proprio impossibile!!
Quindi il risultato della divisione va arrotondato al numero intero superiore.
Nella figura n. 1 possiamo capire più facilmente il ragionamento del come il programma esegue il compattamento di un numero intero positivo.
Cosa dobbiamo fare se il numero intero preso in esame risultasse negativo?
Quando, dopo opportuno test, ho riscontrato la negatività del numero eseguo una somma sul primo valore (byte) del compattato, cioè aggiungo 128.
Vediamo infatti nella figura n. 2 che il primo byte della stringa del compattato ha un valore decimale 140 che corrisponde alla somma di 12 (prime due cifre del nostro numero) e di 128 (convenzione di negatività).

Figura n. 1
Impaccamento (computational -3) di un numero positivo di n. cifre.
Nell'esempio il numero è 1, 2, 3, 4, 5, 6, 7, 8, 9, 0.

Il programma: Lo spacca Byte Numerico

Come si può notare dall'intestazione sia questo programma che quello che verrà in seguito, potrà essere utilizzato e provato su qualsiasi tipo di Commodore, dal VIC20 ai nuovi prodotti.
Essenzialmente il programma si può dividere in tre parti distinte:
Prima parte inerente alla inputazione del numero intero positivo o negativo. In questa fase che può essere distinta dalla riga 340 alla riga 470, si eseguono una serie di controlli formali, in modo tale che effettivamente il dato fornito in input sia effettivamente numerico.
La seconda parte, quella cioè che va dalla riga 530 fino alla riga 700 esegue prima l'esclusione del segno, se c'è, e quindi pareggia la lunghezza in maniera tale che può essere compattato con precisione.
La terza parte infine esegue la ricomposizione e ricostruzione del numero così come era partito dall'input dell'operatore.
La parte finale è stata messa puramente al fine di conoscere le varie caratteristiche delle variabili manipolate e per verificarne l'effettiva riuscita.

Inserire il Listato Lo Spacca Byte Numerico

Sui grossi computers o su quegli elaboratori che utilizzano linguaggi evoluti non è possibile compattare una stringa alfanumerica dichiarata come tale ed il cui contenuto non sia solamente numerico.
In BASIC però possiamo farlo!
È chiaro che devo scendere un po' a compromessi rispetto al compattamento numerico.
Innanzitutto il risultato di una compattazione di alfanumerico non potrà essere di lunghezza esattamente la metà dell'originale, ma la riduzione corrisponderà ai due terzi della lunghezza multipla di tre che contenga la stringa stessa.
Infatti da una stringa di tre caratteri compatto ad una nuova stringa di lunghezza due.
Seconda limitazione: i caratteri che possono essere compattati corrispondono solo ai numeri, tutte le lettere dell'alfabeto e a quattro simboli (lo spazio, il meno, il punto e la barra di divisione) per un totale di quaranta caratteri, mentre tutti gli altri vengono trasformati in spazio.
Perché solo quei simboli grafici?
I tre simboli grafici (—,.,/), lo spazio è implicito, sono facilmente ritrovabili nelle stringhe, infatti il — o la / dividono normalmente una data, mentre il punto chiude un perio-

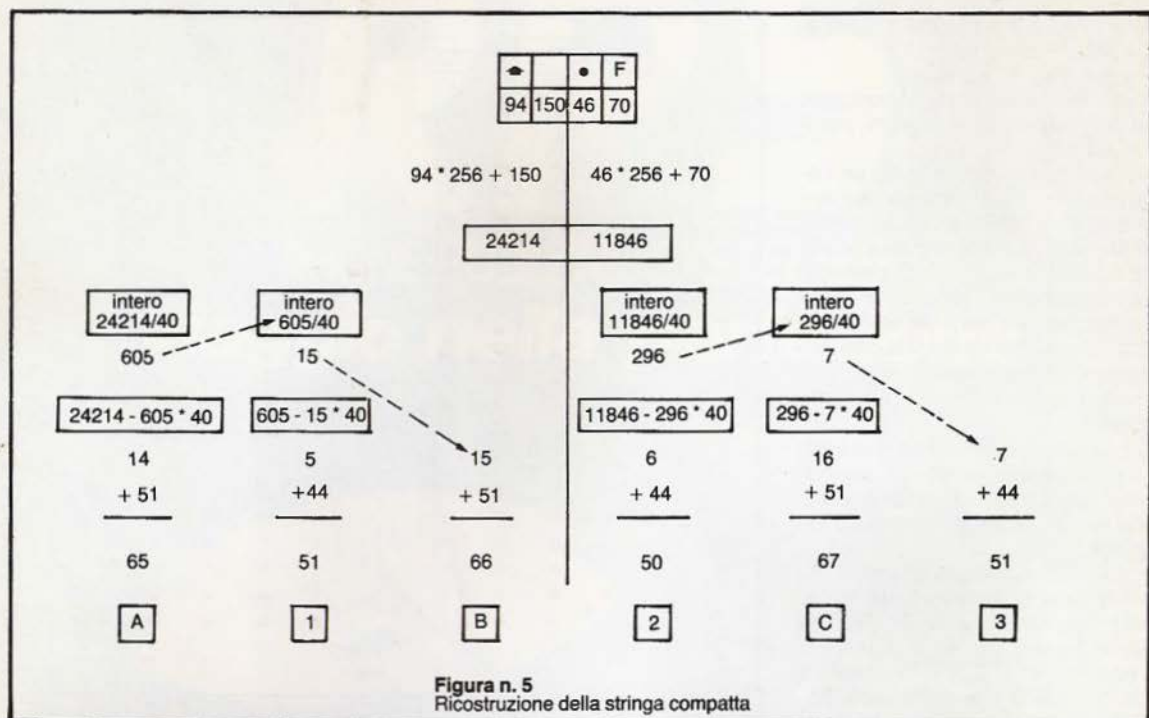
Figura n. 2
Impaccamento (computational -3) di un numero negativo in cifre.
Nell'esempio il numero è - 1, 2, 3, 4, 5, 6, 7, 8, 9, 0.

```
100 REM ****************************
110 REM *                          *
120 REM *     LO SPACCA BYTE       *
130 REM *                          *
140 REM *                          *
150 REM *        NUMERICO          *
160 REM *                          *
170 REM ****************************
180 REM *                          *
190 REM * AUTHOR SOFTWARE :        *
200 REM *                          *
210 REM *  GLORIANO ROSSI (I2KH)   *
220 REM *                          *
230 REM ****************************
240 REM * VIC 20             SI    *
250 REM * COMMODORE 64       SI    *
260 REM * COMMODORE 4000     SI    *
270 REM * COMMODORE 8000     SI    *
280 REM * COMMODORE 16       SI    *
290 REM * COMMODORE 116      SI    *
300 REM * COMMODORE 264      SI    *
310 REM * COMMODORE 364      SI    *
320 REM ****************************
330 :
340 PRINT"[CLEAR][ROSSO]BATTI UN N
    UMERO DI N CIFRE[NERO]"
350 INPUT XX$
360 L=LEN(XX$)
370 :
380 REM ****************************
390 REM * CONTROLLO FORMALE        *
400 REM ****************************
410 :
420 ER=0:FOR N=1 TO L
430 :X$=MID$(XX$,N,1)
440 :IF N=1 AND X$="-" THEN NE=-1:
    GOTO 460:REM  SE NEGATIVO
450 :IF X$<"0" OR X$>"9" THEN ER=E
    R+1
460 NEXT N
470 IF ER THEN PRINT"CI SONO "ER"
    ERRORI NEL NUMERO!!!":PRINT"RI
    PETI":GOTO 340
480 :
490 REM ****************************
500 REM * TOGLIE IL SEGNO          *
510 REM ****************************
520 :
530 IF NE THEN XX$=MID$(XX$,2):L=L
    -1
540 :
550 REM ****************************
560 REM * PAREGGIA LA LUNGHEZZA    *
570 REM ****************************
580 :
590 IF (L/2)<>INT(L/2) THEN XX$="0
    "+XX$:L=L+1
600 :
610 REM ****************************
620 REM * COMPUTATIONAL 3          *
630 REM ****************************
640 FOR I=1 TO L STEP 2
650 :C=VAL(MID$(XX$,I,2))
660 :IF NE AND I=1 THEN C=C+128:RE
    M  SE NEGATIVO
670
680 :C$=CHR$(C)
690 :C3$=C3$+C$
700 NEXT
710 :
720 REM ****************************
730 REM * RICOMPOSIZIONE DI COMP3*
740 REM ****************************
750 :
760 C$="":NE=0
770 FOR I=1 TO LEN(C3$)
780 :C=ASC(MID$(C3$,I,1))
790 :IF I=1 AND C>128 THEN C=C-128
    :NE=-1:REM  RICONOSCE SE ERA N
    EGATIVO
800 :C$=STR$(C)
810 :C$="00"+RIGHT$(C$,LEN(C$)-1)
820 :C$=RIGHT$(C$,2)
830 :IF I=1 AND LEFT$(C$,1)="0" TH
    EN C$=RIGHT$(C$,1)
840 :IF I=1 AND NE THEN C$="-"+C$:
    REM  SE NEGATIVO
850 :CC$=CC$+C$
860 NEXT I
870 :
880 REM ****************************
890 REM * DISPLAY DI CONOSCENZA    *
900 REM ****************************
910 :
920 PRINT"[2 DOWN][BLEU]LUNGHEZZA
      [NERO]"L
930 PRINT"[BLEU]N.INIZIALE  [NERO]
    "VAL(XX$)*NE
940 PRINT"[BLEU]COMP  3     [NERO]
    "C3$
950 PRINT"[BLEU]LEN DI C3$  [NERO]
    "LEN(C3$)
960 PRINT"[BLEU]RINORMALIZ. [NERO]
    "CC$
```

do o è facente parte della punteggiatura di un numero.
Perché solo 40 caratteri?
Beh... qui la faccenda si complica un po' e spero di spiegarmi per il meglio.
Noi sappiamo che un byte, un carattere, può contenere un numero che va da 0 a 255, cioè 256 numeri. Due bytes mi danno una scelta da 0 a 255 e da 0 a 255, cioè 256 e 256 possibilità, quindi se moltiplico il primo byte per il secondo avrò una scelta che va da 0 a 65536.
Dentro questo numero devo farci stare la codifica di tre caratteri originali.
Quale range, quale numero di scelte, mi permette, tramite un determinato algoritmo, di rimanere nel 65536 e comunque avere tre codifiche?
Il numero incriminato è proprio 40. Infatti se io moltiplico 40 per 40 per 40 mi dà esattamente 64.000. Ma allora perché non 41 o più?
Vediamo: 41 × 41 × 41 è uguale a 68.921, quindi non ci stà in 65536.
Se attribuisco un numero che va da 0 a 39 per ogni carattere previsto ho a questo punto una tabellina di conversione.
Infatti per facilitare le cose parto dal valore ASCII dei numeri e delle lettere; se sono numero (compreso il —, il . e la /) tolgo al valore ASCII la costante 44, mentre se è una lettera tolgo 51 (vedi tabella riportata in Figura n. 3).
Dai tre valori così calcolati, moltiplicando il primo per 1, il secondo per 40 ed il terzo per 1600 (40 × 40) ed eseguendo la somma dei valori trovati ottengo un numero che non potrà essere mai maggiore di 64000, il quale scomposto in due bytes mi da due caratteri del codice ASCII memorizzabili sia in memoria RAM che in memoria di massa.
Per meglio capire il ragionamento suggerisco di rileggere le righe precedenti osservando la figura n. 4
La ricostruzione della stringa a due in stringa originale a tre ci porta ad un nuovo algoritmo di relativa complessità se si è capita esattamente la logica di compattazione.
Dal valore ASCII dei due caratteri mi riporto al valore compreso fra 0 e 64000, moltiplicando il primo per 256 e sommando il secondo.
Da questo numero diviso per 40 considero solo la parte intera che moltiplicata per 40 e sottratta dal numero originale mi dà un valore a cui devo aggiungere 51 se è superiore a 13 (ricostruzione di lettera), mentre devo aggiungere 44 se è inferiore di 14 (ricostruzione di numero).
Dal numero trovato dall'intero della divisione del valore iniziale fratto 40 ripeto la medesima operazione per la seconda lettera.
Dall'intero del secondo numero fratto 40 ho direttamente il terzo nu-

carattere	ASCII	COSTANTE	NUMERO ATTRIBUITO
/	32		0
-	45	-44	1
.	46	-44	2
/	47	-44	3
0	48	-44	4
1	49	-44	5
2	50	-44	6
3	51	-44	7
4	52	-44	8
5	53	-44	9
6	54	-44	10
7	55	-44	11
8	56	-44	12
9	57	-44	13
A	65	-51	14
B	66	-51	15
C	67	-51	16
D	68	-51	17
E	69	-51	18
F	70	-51	19
G	71	-51	20
H	72	-51	21
I	73	-51	22
J	74	-51	23
K	75	-51	24
L	76	-51	25
M	77	-51	26
N	78	-51	27
O	79	-51	28
P	80	-51	29
Q	81	-51	30
R	82	-51	31
S	83	-51	32
T	84	-51	33
U	85	-51	34
V	86	-51	35
W	87	-51	36
X	88	-51	37
Y	89	-51	38
Z	90	-51	39

Figura n. 3

Figura n. 4
Fase di compattamento della divisa A1B2C3

Figura n. 5
Ricostruzione della stringa compatta

mero per la terza lettera.
Tutto qui (In ogni caso ritornate sul ragionamento osservando la figura n. 5).

Il listato del programma: "Lo Spacca Byte Alfa" risulta essere un po' più lungo del primo, un po' per la complessità dell'argomento ed un po' perché ho voluto commentarlo meglio con le REMarcks.

Una descrizione, quindi, particolareggiata, riga per riga come è mia abitudine, potrebbe in questo caso confondere più che chiarificare.

Un unico commento prima di concludere: in entrambi i listati potrete notare che i cicli di FOR... NEXT sono evidenziati con il carattere ":" in inizio riga. È solo un fatto estetico che ci permette a colpo d'occhio di individuare i vari loop, dove iniziano e dove finiscono.

Tenete presente che se li battete avrete due listati esteticamente belli, a discapito però della velocità, infatti l'interprete BASIC deve perdere tempo per riconoscere il ":" e non fare nulla, come pure perde tempo per riconoscere le REM e saltare alle seguenti istruzioni.

```
100 REM ***************************
105 REM *                         *
110 REM * LO SPACCA BYTE ALFA     *
115 REM *                         *
120 REM *     IMPACCAMENTO        *
125 REM *          E              *
130 REM *    DISIMPACCAMENTO      *
135 REM *                         *
140 REM ***************************
145 REM *                         *
150 REM * AUTHOR SOFTWARE :       *
155 REM *                         *
160 REM *  GLORIANO ROSSI (I2KH)  *
165 REM *                         *
170 REM ***************************
175 REM * VIC 20              SI  *
180 REM * COMMODORE 64        SI  *
185 REM * COMMODORE 4000      SI  *
190 REM * COMMODORE 8000      SI  *
195 REM * COMMODORE 16        SI  *
200 REM * COMMODORE 116       SI  *
205 REM * COMMODORE 264       SI  *
210 REM * COMMODORE 364       SI  *
215 REM ***************************
220 :
225 PRINT"[CLEAR][2 DOWN][ROSSO]BATTI UNA STRINGA DI CARATTERI[NERO]"
230 INPUT "[NERO]A$=[VERDE]";A$
```

```
235 XA$=A$:GOSUB 325
240 GOSUB 540:B$=XA$
245 PRINT"[2 DOWN][VERDE]A$=[NERO]
    ";A$
250 PRINT"[DOWN][RVS][NERO]LEN PRI
    MA =[VERDE]";LEN(XA$)
255 PRINT"[DOWN][RVS][NERO]LEN DOP
    O  =[VERDE]";LEN(XX$)
260 PRINT"[2 DOWN][BLEU]ZZ$=";XX$
265 PRINT"[2 DOWN][VERDE]B$=[NERO]
    ";B$
270 PRINT"[DOWN][RVS][NERO]LEN RIP
    R  =[VERDE]";LEN(XA$)
275 INPUT "[ROSSO]VUOI RIPROVARE (
    S/N)[NERO]";K$
280 IF LEFT$(K$,1)="N" THEN END
285 GOTO 225
290 REM ******************
295 REM * COMPATTAMENTO    *
300 REM *      DI STRINGA  *
305 REM ******************
310 REM * AGGIUSTAMENTO    *
315 REM * LUNGHEZZA  XA$   *
320 REM ******************
325 X3=INT((LEN(XA$)+2)/3)*3-LEN(X
    A$)
330.IF X3=0 THEN 345:REM  E' UN MU
    LTIPLO DI 3
335 IF X3=1 THEN XA$=XA$+CHR$(0):G
    OTO 345:REM  SE MANCA 1 CHR
340 XA$=XA$+CHR$(0)+CHR$(0):REM  S
    E MANCANO 2 CHR
345 X3=LEN(XA$)
350 XX$=""
355 REM ******************
360 REM * PRENDE IN ESAME *
365 REM * 3 CHR PER VOLTA *
370 REM ******************
375 FOR XI=1 TO X3 STEP 3
380 :X3$=MID$(XA$,XI,3):XS=0
385 :REM ******************
390 :REM * ESAME DI 1 CHR  *
395 :REM * DEI 3,PER VOLTA *
400 :REM ******************
405 :FOR XJ=1 TO 3
410 ::XK=ASC(MID$(X3$,XJ,1))
415 ::IF XK<45 THEN XK=0:GOTO 485
420 ::REM ****************
425 ::REM * SE E' :        *
430 ::REM * -./0123456789 *
435 ::REM ****************
440 ::IF XK<58 THEN XK=XK-44:GOTO
    485
445 ::IF XK<65 THEN XK=0:GOTO 485
450 ::REM ****************
455 ::REM * SE E' :        *
460 ::REM * ABCDEFGHIJKLM *
465 ::REM * NOPQRSTUVWXYZ *
470 ::REM ****************
475 ::IF XK<91 THEN XK=XK-51:GOTO
    485
480 ::XK=0
485 ::XS=XS+XK*(40↑(XJ-1))
490 :NEXT
495 :XK=INT(XS/256)
500 :XJ=XS-XK*256
505 :XX$=XX$+CHR$(XK)+CHR$(XJ)
510 NEXT
515 RETURN
520 REM ******************
525 REM * RICOSTRUZIONE    *
530 REM *      DI STRINGA  *
535 REM ******************
540 XA$=""
545 FOR XI=1 TO LEN(XX$) STEP 2
550 :XK=ASC(MID$(XX$,XI,1))
555 :XJ=ASC(MID$(XX$,XI+1,1))
560 :XK=XK*256+XJ
565 :REM ******************
570 :REM * PRIMO CARATTERE *
575 :REM ******************
580 :XJ=INT(XK/40):X3=XK-XJ*40:GOS
    UB 670
585 :XA$=XA$+CHR$(X3)
590 :REM ******************
595 :REM * SECONDO CHR     *
600 :REM ******************
605 :XK=INT(XJ/40):X3=XJ-XK*40:GOS
    UB 670
610 :XA$=XA$+CHR$(X3)
615 :REM ******************
620 :REM * TERZO CARATTERE *
625 :REM ******************
630 :X3=XK:GOSUB 670
635 :XA$=XA$+CHR$(X3)
640 NEXT
645 RETURN
650 REM ******************
655 REM * RICOSTRUZIONE    *
660 REM * VALORE  ASCII    *
665 REM ******************
670 IF X3=0 THEN X3=32:RETURN
675 IF X3<14 THEN X3=X3+44:RETURN
680 X3=X3+51:RETURN
```

IL SYSTEM RESET

di Tullio Spezia (i 2 TZS)

Chi possiede un pò di pratica di apparecchi elettronici, conosce bene la funzione del RESET, cioè quella funzione utile per riportare una macchina nelle condizioni iniziali di lavoro. Normalmente questa funzione è attivata da quel pulsantino che mette di solito a massa un punto del circuito. Tutto ciò è utilizzabile anche nei Computer, e'...., se non c'è, si può sempre metterlo.

Vediamo come lo si può aggiungere ad un Commodore C-64, senza manometterlo, nè tantomeno dare adito a danni presenti e futuri dell'apparecchio, ed impariamo anche la sua importante funzione e comodità.

Se fosse solo per la comodità .... diciamo che si fa prima a schiacciare un pulsantino che a digitare sulla tastiera il famoso SYS 64738 per ricominciare con un nuovo programma: tanto più che bisognerebbe ricordare a memoria il "64738". Altrimenti bisogna andarselo a cercare da qualche parte, per non parlare di come certi programmi si "piantano" in maniera tale che nè lo STOP e nemmeno lo STOP e RESTORE riescono a farli tornare alla normalità.

Nulla varranno le nostre disperate "pestate" sui tasti, nè tanto meno si potrà spegnere senza perdere il lavoro fatto con tanta pazienza poco prima.

Ecco come risolvere per il meglio il pasticciaccio: azionando il RESET così chiamiamo il pulsantino che vi propongo si avrà la restituzione dell'uso della tastiera. Ciò consente di effettuare il LOAD di un particolare programmino ausiliario (in linguaggio macchina) con cui recuperiamo prontamente il programma in RAM che sembrava perduto. E se

per caso, si fa per dire, sbadatamente abbiamo digitato NEW e fatto il RETURN (un caso tipico da pistola alla tempia e..... CLICK ...PUMM) Con un pò di freddezza è possibile azionare il RESET, fare il LOAD del programmino ausiliario, ed ecco presente al LIST od al RUN il programma a cui avevamo dato, con il NEW, anche i nostri saluti.

Intanto, diamo una buona occhiata al manuale d'uso del c-64, o meglio ancora alla Reference Guide. Andiamo alla pagina in cui si parla della USER PORT, cioè quel connettore che dovrebbe essere utilizzato per gli apparecchi esterni.

É chiaro che dovremmo aver acquistato un connettore adatto, completo di guscio di protezione, e dotato d'un pulsantino che fisseremo sul guscio stesso. Facendo il collegamento del pulsantino alle pagliette del connettore stesso avremo tutto quanto è necessario per avere il RESET senza manomettere o aprire il C-64.

Osserviamo il manuale: il piedino.3 è appunto il reset, ed il piedino 1 corrisponde alla massa del circuito. Visto? Sono quelle le due pagliette del connettore a cui devono essere saldati i due conduttori che vanno al pulsantino. Ma, se non siete sicuri delle vostre capacità di saldatura è bene far eseguire la realizzazione pratica a qualsiasi buon tecnico del saldatore anche non del campo computer.

Non fate gli impazienti. Quindi non toccate sbrigativamente gli stessi punti suggeriti usando una forcina da capelli; ciò facendo, avendo la grande probabilità di fare qualche cortocircuito poco gradito dall'apparecchio.

Cosa accade utilizzando il RESET?

La Reference Guide, in maniera un pò più ampia, spiega che mettendo provvissorian.ente a massa il piedino 3, si riesce a resettare completamente il C-64, e quindi anche i puntatori del programma in BASIC, mentre la memoria RAM non viene vuotata. Perciò occorre solamente rimettere i puntatori al loro posto, e la situazione iniziale è perfettamente ripristinata.

I puntatori indicheranno all'interprete la posizione del programma da ricuperare solo se non verranno distrurbati dal sopraggingere di un nuovo programma in BASIC. Bisogna perciò usare un programma ausiliario in linguaggio macchina preparato prima su nastro o sul disco.

Oltre a ciò questa utility non deve sovrapporsi ad alcun programma in corso, per non "rompere le uova nel paniere".

L'utility occupa pochi bytes, quindi avremo la possibilità di locarla in una isoletta della memoria che parte alla locazione 679 (02A7 in essa) dove non da fastidio, e trova anche spazio sufficiente.

Il listato del programma BASIC che provvede a generare nella giusta locazione l'utility di RESET è riportato qui più avanti. Dopo averlo digitato fatene subito il SAVE.

Quando darete il RUN, è il programma stesso che pone la routine nella giusta locazione e, ciò fatto, esegue il comando di salvataggio dell'utility stessa, chiamandola proprio RESET. Troveremo quindi automaticamente sul nostro disco (o cassetta) il programma "RESET" vero e proprio.

Sarà di quest'ultimo che faremo il LOAD "RESET" 8, 1 dal disco (oppure LOAD "RESET", 1, 1 se da cassetta) facendo attenzione al "virgola uno" in finale perchè è essenziale.

Ecco il LIST del programma generatore scritto in BASIC:

```
 10 REM : PROGR.  "RESET/PREP"
 20 A=679
 30 READ T: IF T=256 THEN 50
 40 POKE A,T: A=A+1: GOTO 30
 50 POKE 43,679 AND 255: POKE 44,2
 60 POKE 45,733 AND 255: POKE 46,2
 70 CLR: SAVE"0:RESET",8
 80 DATA 160,3,200,177,43,208,251,
    200,200,152,160,0,145,43
 90 DATA 165,44,200,145,43,133,60,
    160,0,132,59,162,0,200
100 DATA 208,2,230,60,177,59,208,2
    45,232,224,3,208,242,200
110 DATA 208,2,230,60,132,45,164,6
    0,132,46,96,256
```

Se non possedete l'unità a dischi, ma solamente il registratore, dovrete sostituire la riga 70 in questo modo:
70 CLR: SAVE "RESET"

Il collaudo

Il collaudo di tutta la faccenda è presto fatto:

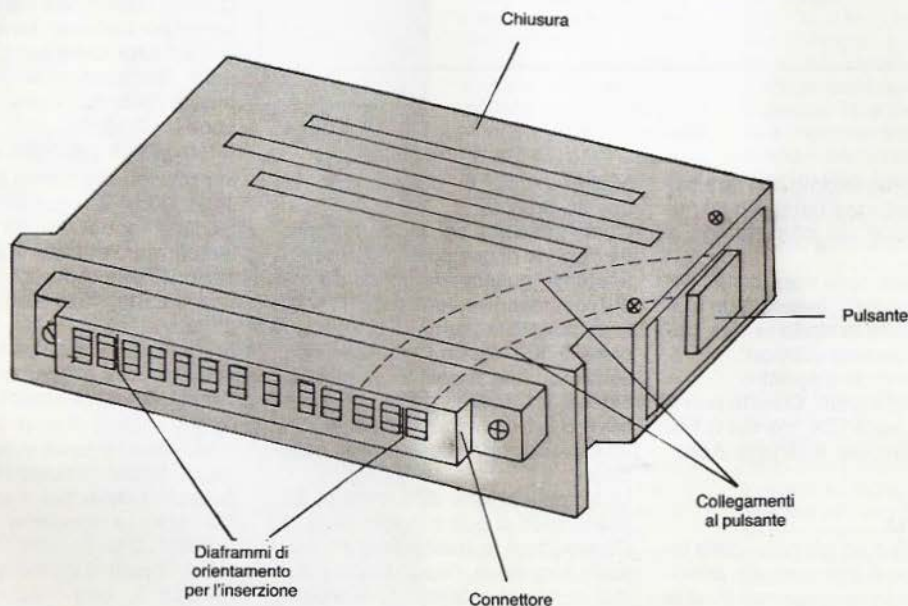
- digitate un qualsiasi programma di poche linee o caricatelo dal disco o cassetta;
- eseguite il NEW;
- premete brevemente il pulsantino di RESET;
- poi: LOAD "RESET", 8,1, (ricordare il "virgola uno"; e per la cassetta LOAD "reset", 1,1);

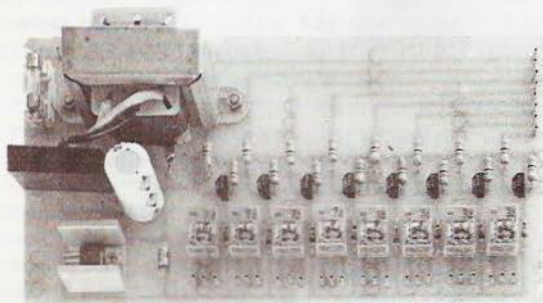
e dopo il RETUR, digitate SYS 679 per avviare il programma in linguaggio macchina.

Quando riappare il cursore, in pratica quasi istantaneamente, potete procedere col LIST (o col RUN) e si avrà la gradita ricomparsa del programma che avevate annientato col NEW.

Due parole sul connettore: può essere utilizzato il tipo TRW 251-12-50-170/50-24SN-9 assieme al copriconnettore della Cannon tipo DD 115 339-4, avvitandolo con due viti poco più lunghe di quelle fornite. Farsi dare (o rimediare) due divisori da inserire opportunamente nel connettore TRW per impedire una inserzione capovolta, e che devono corrispondere agli appositi intagli presenti sulla piastra del C-64.

Una volta costruito questo piccolo accessorio può essere lasciato sempre innestato nell'incavo della USER PORT, alla quale si adatta perfettamente.

Pulsante di Reset per C-64

C.I.A.

di Ernesto Sidoti e Renzo Zonin

John si guardò intorno con fare circospetto, estrasse dal tacco un microfilm e fece per consegnarlo a Bors.
Fermi, restate dove siete, siamo del controspionaggio, gracchiò un megafono mentre la notte veniva illuminata da potenti riflettori...
Fermo non girar pagina!
Abbiamo scherzato. Questo non è un articolo sulla CIA, ma sul C.I.A. (ovvero Complex Interface Adapter).

Cos'è il CIA

Il CIA (6526) è un integrato della famiglia 65xx. Il suo compito principale riguarda la gestione di due porte ingresso uscita.
Nel Commodore 64 sono montati due di questi versatili integrati, uno è utilizzato dal sistema per gestire la tastiera, l'altro per comunicare con sistemi esterni, proprio di quest'ultimo chip ci occuperemo in questo articolo.
Come forse non sai, il Commodore 64 dispone di una potentissima porta utente, questa supportata da pochi componenti esterni e da un software adeguato permette di interfacciare il tuo Home-Computer con estrema facilità, così il tuo piccolo calcolatore avrà la possibilità di ricevere o trasmettere informazioni all'esterno senza l'ausilio dell'operatore.
Una applicazione di tale porta ad esempio la si può avere in casa. Il Commodore potrebbe farti da versatile segretaria, rispondendo automaticamente al telefono, mandando un messaggio preregistrato o magari sintetizzando la voce e avviando nel contempo un registratore capace di memorizzare il messaggio dell'interlocutore. Così quando tornerai a casa potrai ascoltare le telefonate pervenute in tua assenza.
Quando sei in casa potrebbe telefonare automaticamente e ripetere il numero da te scelto fin tanto che non risulta libero. Un'altra applicazione potrebbe essere quella di antifurto. Il computer collegato al sistema di allarme le luci di casa, la televisione o la radio simulando la presenza di gente.
Se scatta l'allarme lui telefona a dei numeri da te dati prima di uscire avvertendo così del verificato allarme.
Nel campo del lavoro le applicazioni sono tante, ad esempio l'automazione delle pompe da giardino. Queste, comandate dal calcolatore, verranno azionate tenendo conto dell'umidità della terra o dell'aria della temperatura e della luce e inoltre del tipo di coltura e della disponibilità idrica.
Interessante potrebbe essere una applicazione in campo fotografico. Il tuo Home adeguatamente interfacciato potrebbe controllare la temperatura dei bagni, la temporizzazione delle esposizioni in base al tipo di carta e alla trasparenza del negativo.
Un'applicazione ideale la si ha in discoteca dove potrebbe controllare l'impianto di illuminazione.
Spero adesso di aver acceso una costruttiva fantasia in te e di averti un po' fatto conoscere le infinite applicazioni della tua macchina.
Ma adesso torniamo alla porta utente. Da questo connettore (12 + 12 poli) abbiamo a disposizione ben 8 porte input-output programmabili singolarmente e un interfaccia RS232 programmabile, ma purtroppo non standard per quanto riguarda le tensioni di ingresso e di uscita (poco male con poca circuiteria diventa standard).

In questo numero ci occuperemo proprio della porta di input-output. Per i principianti, le porte sono dispositi che permettono ai dati di entrare o uscire dal calcolatore. La porta viene detta seriale se i dati passano bit dopo bit in successione attraverso un solo filo, un esempio di questo tipo di porta è l'interfaccia per il tape. Una porta invece si dice parallela se i dati passano Byte dopo Byte cioè gli otto Bit che rappresentano il dato si presentano contemporaneamente sulle gli otto pin della porta. La porta B del tuo Commodore 64 è proprio una porta parallela. I pin C D E F H J K L (vedi fig. 1) sono collegati direttamente ai piedini dell'integrato 6526.

Da ognuno di questi pin possono entrare o uscire tensioni a secondo se il pin è stato settato in ingresso o uscita.

Il registro che decide quale Bit settare in uscita e quale in ingresso è il D.D.R. (Data Direction Register). Questo registro risiede nella locazione 56579.

Mettendo 1 in un Bit del D.D.R. si ha il corrispondente pin settato in uscita, cioè si avr« la possibilità di disporre sul pin di una tensione bassa o alta secondo programma. Settando a zero un Bit del D.D.R. la porta sarà pronta a ricevere dati dall'esterno ossia nel corrispondente pin potremo applicare una tensione bassa o alta secondo necessità.

Quindi con POKE 56579,255 la porta sarà settata tutta in uscita, con POKE 56579,0 tutta in ingresso. Con POKE 56579,128 si mette in uscita il Bit 7, i restanti 7 bit in ingresso (vedi fig. 2).

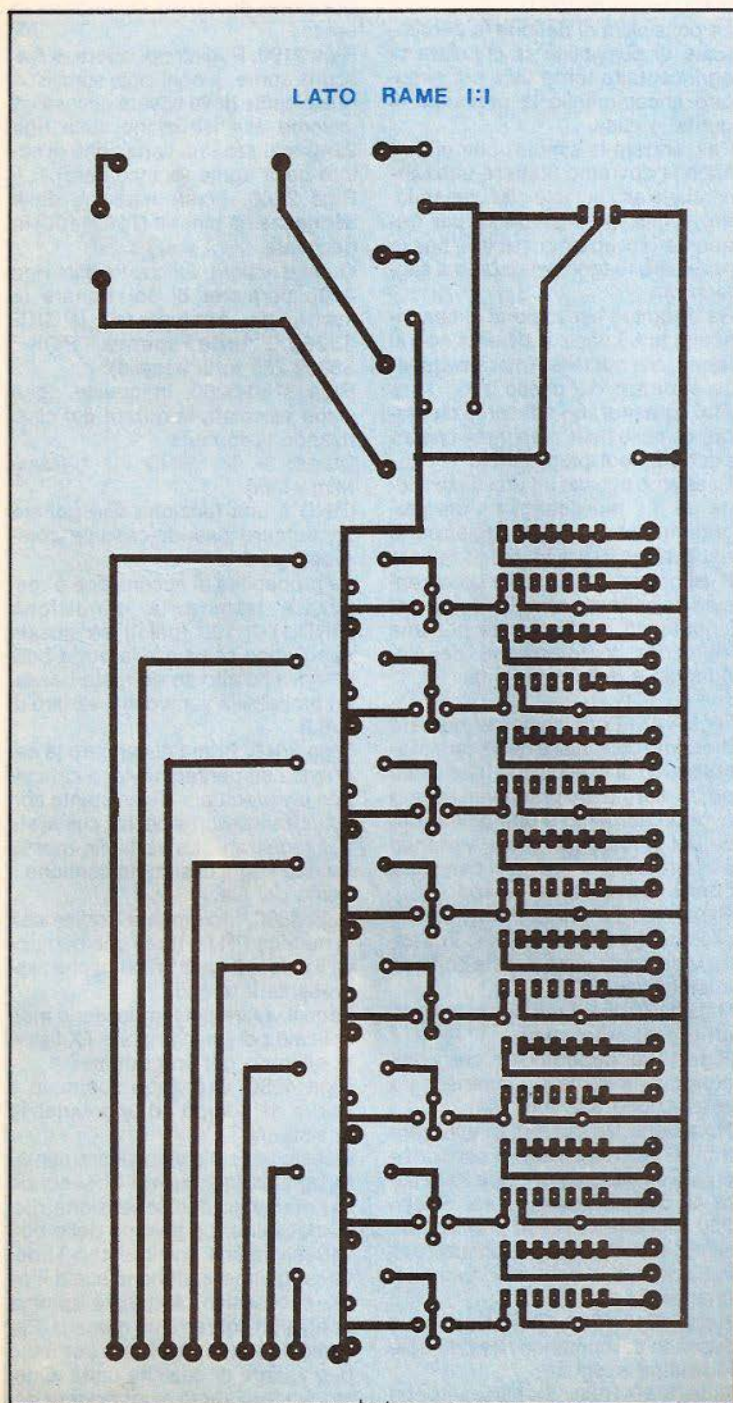
Dopo aver fissato quale Bit deve ricevere e quale deve trasmettere, bisogna adesso leggere o scrivere su tale porta. Questa funzione è svolta dal registro P.R.B. (Peripheral Data Reg B) locato in 56577.

Supponiamo adesso di aver messo tutti i bit del D.D.R. a 0 cioè la porta è pronta a ricevere per ogni singolo ingresso due valori di tensione, uno basso che rappresenta lo 0 l'altro alto rappresentante l'uno.

Se agli otto ingressi adesso appli-

chiamo una tensione alta e contemporaneamente leggiamo il valore del P.R.B. (PEEK(56576)) il calcolatore ci ritornerà il valore 255 indicandoci l'avvenuta commutazione in valore alto.
Se invece leggiamo il valore 253 significherà che avremo una tensione bassa nei pin che fanno capo ai BIT 0 e 1 del P.B.R. una tensione alta nei restanti pin.
Allo stesso modo se tutte le porte sono settate in uscita (POKE 56579,255) quando nel P.R.B. verrà messo il valore 255 sugli otto pin interessati della porta utente si potrà leggere con uno strumento una tensione alta, circa 5 Volt, se nella locazione 56577 verrà messo il valore 253 si misurerà una tensione bassa nei pin collegati ai primi due bit del P.R.B. alta nei restanti. Ovviamente potremo settare ad esempio i primi 4 Bit in ingresso i rimanenti in uscita e compiere regolarmente le operazioni di scrittura e lettura, ricordandoci di scrivere nel P.R.B. solo valori atti a pilotare i BIT 4 5 6 7.
Questo mese presentiamo una semplice e funzionale scheda capace di aprire e chiudere degli interruttori e il relativo software per pilotarla.

L'Hardware

La scheda è stata realizzata con pochissimi componenti tutti di facile reperibilità e basso costo.
L'hardware è composto da otto moduli identici e di un modulo alimentazione, tutti i componenti compreso il trasformatore di alimentazione sono montati su di una basetta di 19,5 × 10 mm. Le uscite del tuo 64 verranno collegate ognuno a altrettanti ingressi della scheda. Quando in uno degli 8 pin (C D,E F H J K L) si presenterà una tensione alta il transistor che è collegato a quel pin farà da interruttore permettendo l'eccitazione del relè e la conseguente chiusura del contatto.
I transistor sono dei BC 147A nessuno vieta però di utilizzare altri transistor simili.
Qualche parola va spesa per i relè. Noi in questa scheda abbiamo montato dei relè di bassa potenza

chi volesse commutare carichi di alto assorbimento non dovrà far altro che collegare a ogni relè un relè aggiuntivo rispondente alle caratteristiche di assorbimento richieste. La bobina dei relè che monterai nella scheda non dovrà essere di valore inferiore a 50 OHM, meglio se il valore è di 90 OHM.

Collaudo della scheda

Dopo aver montato tutti i componenti guarda attentamente se per caso non hai commesso errori.
Attenzione alla polarità dei diodi e dei transistor.
Se tutto è montato al posto giusto e nel verso giusto puoi iniziare il collaudo. Prendi un tester e dopo aver dato tenzione alla scheda controlla se su C1 vi siano 5 Volt dopo di ciò prendi uno spezzone di filo di rame e collegalo al polo positivo della alimentazione, adesso appoggialo sulle entrate della scheda. I relè dovrebbero eccitarsi se così non è spegni tutto e ricontrolla, forse i transistor sono stati montati al rovescio.
Quando tutto funziona collega il connettore, attento a collegare i fili al posto esatto, e avvia le prove con il 64.

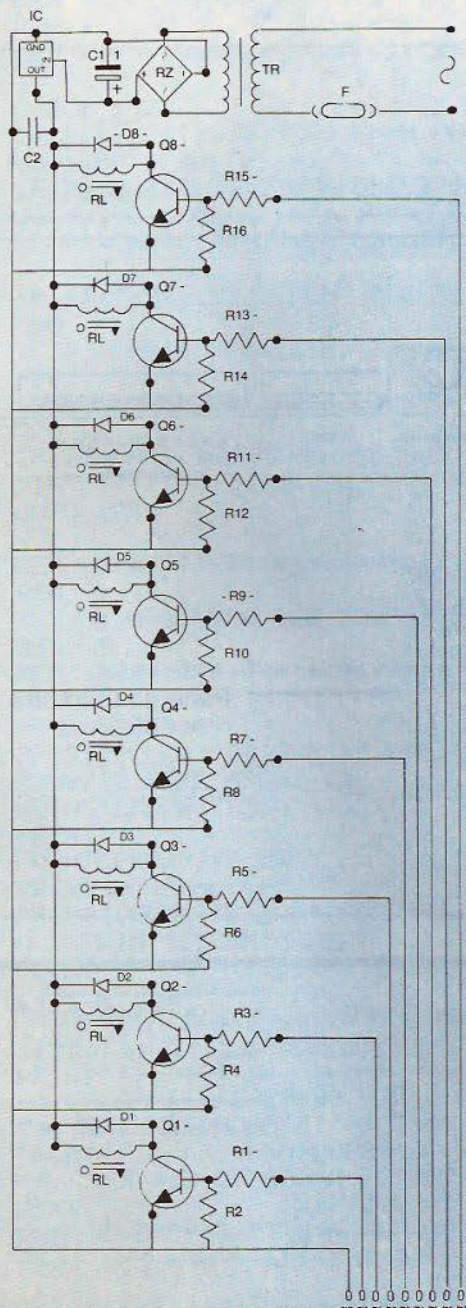
Attenzione:

Non collegare mai la scheda quando il computer è acceso.

Il Software

Il programma di gestione è strutturato a moduli; questo sia per permettere una maggiore leggibilità sia per facilitare eventuali modifiche e aggiunte.
Tutto il programma ruota attorno ad una piccola procedura (MAIN) che, a sua volta secondo i comandi ricevuti dall'utente, richiama tante piccole subroutine.
Appena lanci il programma (RUN) il 64 visualizzerà un menù con tutte le opzioni offerte.
Molto probabilmente userai la prima "programmazione". Dopo aver premuto F1 appariranno 8 caselle di eguale grandezza, numerate da 0 a 7, e una lampadina lampeggerà nella casella 0. Il lampeggio sta

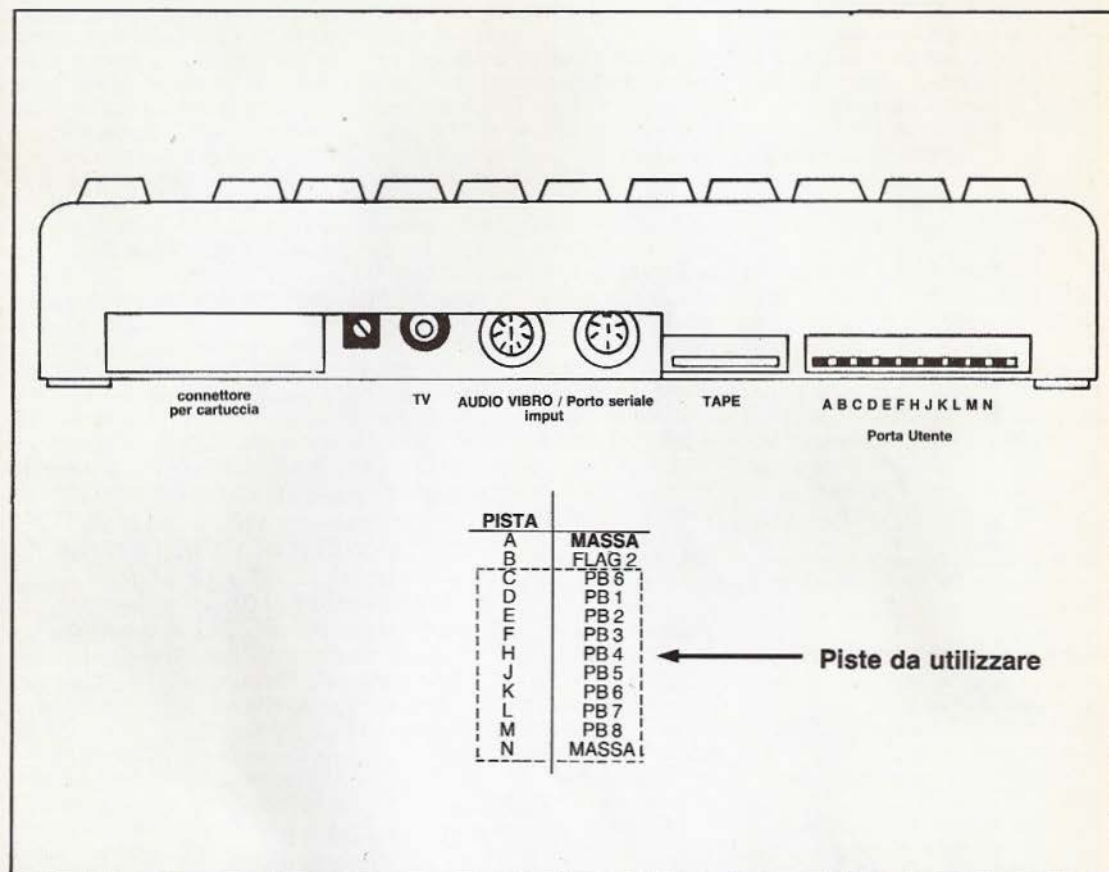
ad indicare che il computer aspetta il comando. Ognuna di queste otto caselle raffigura una porta del tuo 64, la lampadinà accesa nella rispettiva cella, indicherà la chiusura del rispettivo interruttore nella scheda in fase di esecuzione. Se adesso premi Return la lampada resterà accesa nella prima casella e passerà alla posizione successiva, se premi la barra invece passerà solo alla casella successiva.
Dopo aver definito il passo non resterà che definire il tempo per cui dovrà restare attiva questa configurazione.
L'unità di misura è il secondo. Il valore verrà impostato digitando sui tasti numerici, verranno accettati valori non superiori a 999999 secondi. Se hai necessità di avere unità di misura diverse non dovrai far altro che diminuire o aumentare il valore di TX. Questa variabile è definita nella prima riga di programma.
Se sei soddisfatto del passo già definito premi semplicemente Return passerai alla definizione del successivo. Se invece non ti piace premi space e rimposta tutto.
Quando il numero dei passi è sufficiente per i tuoi scopi premi la freccia che punta a sinistra e ritornerai al menù.
Le altre opzioni ti permetteranno di rivedere e correggere i passi già fatti, registrarli e rileggerli e infine di creare automaticamente una sequenza casuale.
Quest'ultima opzione si presta a molteplici usi. Potresti generare una sequenza che ti accenda casualmente le luci di casa o casualmente accenda i faretti della tua discoteca ecc. Premendo F6 il 64 ti chiederà il tempo minimo e massimo che dovrà assumere ogni passo e la percentuale di probabilità di chiusura di ogni contatto. Se ad esempio nella casella 0 metti la probabilità a 100 la lampada resterà sempre accesa per tutti i cicli. Viceversa se la probabilità è 0 resterà sempre spenta. Ovviamente questi sono i due casi limite la percentuale può essere qualunque basta che sia compresa tra 0 e 100, valori maggiori verranno considerati 100 valori minori 0.
La possibilità di definire la percentuale di possibilità di chiusura di ogni contatto torna utile per simulare ancor meglio la presenza di gente in casa.
Per rendere la simulazione più affidabile dovremo mettere una percentuale alta sui relè che comandano le luci degli ambienti più frequentati (soggiorno, cucina) bassa nelle altre (bagno, sgabuzzino, soggiorno).
Se durante l'esecuzione si esauriscono tutti i passi il sistema non si ferma, ma ripeterà l'intera sequenza partendo dal passo 0.
Altri comandi sono descritti chiaramente nelle belle maschere che arricchiscono il programma.
Passiamo adesso ad una descrizione un po' più dettagliata del programma, che per la maggior parte risulterà noiosa e inutile in quanto il programma è di facile comprensione e estrema semplicità, ma per i rimanenti risulterà utile per una maggiore comprensione del linguaggio e della macchina.

Riga 1120 Definizione del numero massimo dei passi e del fattore moltiplicativo di esecuzione. Cambiando il valore della prima variabile si avrà un aumento o una diminuzione del numero dei passi, variando il valore della seconda si cambierà l'unità di misura del tempo.
Riga 1125 La prima POKE stabilisce che tutte le porte siano in uscita, la seconda setta tutte le porte allo stato basso.
Riga 1126. Realizzo per tutti i tasti un repeat automatico.
Riga 1150. Leggo i data che compongono la sprite e li inserisco tra la locazione 832 e la 894.
Nota bene, se prevedi di utilizzare il tape per registrare le sequenze create dovrai memorizzare il tuo data altrove ad esempio nel blocco 200 allocato da 12800 fino 12863.
Riga 1160. Definisco i puntatori delle 8 sprite in questo caso tutti puntano al blocco 13.
Riga. 1240-1320. Qui si decide a secondo del comando ricevuto quale routine eseguire.
Riga 1590-1680. Stabilisco in che posizione dovranno accendersi le sprite.
Riga 2190. Registro di colore di ciascuna sprite. A ogni ciclo stabilisco quale sprite deve essere accesa ed insieme alla istruzione sulla riga 2240 realizzo una variazione di colore nella sprite (bianco/nero).
Riga 2260. Trasformazione della sequenza da binaria (11001100) in decimale.
Questo valore utilizzato alla riga 2190 permette di selezionare le sprite da accendere. (POKE 53269,0 tutte spente, POKE 53269,255 tutte accese).
Riga 3160-3260. In queste righe viene calcolata la durata del ciclo usando la formula:
Durata = Int (RND (1) * (Max-Min) + Min)
(RND è una funzione che genera un numero pseudo-casuale compreso tra 0 a 1).
La probabilità di accensione è realizzata tramite la condizione (RND(1) * 100 (pa(J)) se questa condizione si avvera la porta passa a livello alto se no resta bassa. La probabilità varia con il variare di PA(J).
Riga 3640. Prima di caricare la sequenza su periferica viene cancellato un eventuale file esistente con il medesimo nome del file che si sta per registrare. La variabile messa tra due segni di somma contiene il nome del file.
Riga 3860. Incrementa l'indice della matrice PR fin tanto che non trova il valore 0 nella colonna che rappresenta il tempo.
Letto il valore del tempo viene moltiplicato per una variabile TX fissata all'inizio del programma.
Riga 4080. Qui viene sommato il valore del tempo ad una variabile di sistema TI.
Questa variabile viene incrementata ogni sessantesimo di secondo dal momento dell'accensione della macchina. La somma delle due variabili indica il valore che TI deve raggiungere affinché scatti il ciclo successivo. A questa somma dobbiamo sottrarre un numero. Per il mio computer è 11, ma per il tuo può variare di qualche unità. Questo numero serve a correggere de-

Elenco componenti
RI - RI6 Resistenza 4100   1/4 watt
DI-D8 - Diodo IN4148 1/4 watt o simile
CI - Condensatore 2200 juF 16 V
C2 - Condensatore 0.01 juF poliestere
Q1 - Q8 - Trasistor BC 147 A o simile
RZ - Raddrizzatore 1 Amp 20V
IC - Integrato juA 7805
TR - Trasformatore 6V IA
RLT - RL_2 - Relè 6V 90-50
F - Fusidile 3 ma

Varie
Portafusibile DA c.s.
Aletta di raffreddamento
Connettore 12 + 12 per C.S.
Bosetta ramata 10 x 20
Trecciola multicolore 9 capi

gli errori apportati dal ritardo dato dalla stampa sul video di alcuni valori.

Con 11 il 64 usato nelle prove apportava un errore di 1/2 decimo di secondo ogni ora.

Riga 4090. Genera un break fintanto che non si eguagli TI al valore fine.

```
1000 REM *************************
1010 REM *                       *
1020 REM *   OUTPUT   PROGRAM    *
1030 REM *                       *
1040 REM *          DI           *
1050 REM *                       *
1060 REM * ERNESTO      RENZO    *
1070 REM *          &            *
1080 REM *   SIDOTI      ZONIN   *
1090 REM *                       *
1100 REM *************************
1110 :
1120 PS=1000:TX=60
1125 POKE 56579,255:POKE 56577,0
1126 POKE 650,128:REM *REPEAT TAST
     I*
1130 DIM PR(PS,1)
1140 CC$="[BIANCO]TIME":C1$="        "
     :PASSO=0
1150 FOR A=832 TO 894:READ B:POKE
     A,B:NEXT
1160 FOR A=0 TO 7:POKE 2040+A,13:N
     EXT
1170 :
1180 REM *************************
1190 REM *                       *
```

```
1200 REM *          MAIN          *
1210 REM *                        *
1220 REM ************************
1230 :
1240 GOSUB 1820
1250 IF D$=CHR$(133) THEN PA=0:GOS
     UB 1400:GOSUB 2130
1260 IF D$=CHR$(134) THEN GOSUB 14
     00:GOSUB 3960
1270 IF D$=CHR$(135) THEN GOSUB 14
     00:GOSUB 2650
1280 IF D$=CHR$(136) THEN  GOSUB 3
     790
1290 IF D$=CHR$(137) THEN  GOSUB 3
     600
1300 IF D$=CHR$(138) THEN  GOSUB 4
     220
1310 IF D$=CHR$(139) THEN  GOSUB 2
     910
1320 GOTO 1240
1330 :
1340 REM  ***********************
1350 REM  *                     *
1360 REM  * MASCHERA CON LAMPADE *
1370 REM  *                     *
1380 REM  ***********************
1390 :
1400 POKE 53269,0
1410 PRINT"[CLEAR][BIANCO]":POKE 5
     3280,12:POKE 53281,12
1420 PRINT"[DOWN]PASSO"
1430 PRINT"──────────"
1440 PRINT"[DOWN]VALORE"
1450 PRINT"──────────"
1460 PRINT"[HOME][BLEU]":FOR A=1 T
     O 3:PRINT"[DOWN]":NEXT
1470 PRINT"┌───┬───┬───┬───┬───┬──
     ─┬───┬───┬───────┐";
1480 PRINT"| 7 | 6 | 5 | 4 | 3 | 2
      | 1 | 0 | "CC$" [BLEU]|";
1490 PRINT"├───┼───┼───┼───┼───┼──
     ─┼───┼───┼───────┤";
1500 FOR A=1 TO 4
1510 PRINT"|   |   |   |   |   |   
      |   |   |       |";
1520 NEXT
1530 PRINT"└───┴───┴───┴───┴───┴──
     ─┴───┴───┴───────┘";
1540 :
1550 REM ********************
1560 REM * POSIZIONE SPRITE *
1570 REM ********************
1580 :
1590 POKE 53277,0:POKE 53271,0:Y=1
     35
1600 POKE 53264,1
1610 POKE 53248,0:POKE 53249,Y
1620 POKE 53250,224:POKE 53251,Y
1630 POKE 53252,192:POKE 53253,Y
1640 POKE 53254,160:POKE 53255,Y
1650 POKE 53256,129:POKE 53257,Y
1660 POKE 53258,96:POKE 53259,Y
1670 POKE 53260,64:POKE 53261,Y
1680 POKE 53262,32:POKE 53263,Y
1690 REM ***********************
1700 REM * COLORE DELLE SPRITE *
1710 REM ***********************
1720 :
1730 FOR XL=0 TO 7:POKE 53287+XL,1
     :NEXT
1740 RETURN
1750 :
1760 REM ********************
1770 REM *                  *
1780 REM *       MENU'      *
1790 REM *                  *
1800 REM ********************
1810 :
1820 PRINT"[BIANCO][CLEAR]":POKE 5
     3280,0:POKE 53281,0
1830 PRINT"┌──────────────────────
     ───────────────┐";
1840 PRINT"|PROGRAMMA GESTIONE POR
     TE DI USCITA (C)|";
1850 PRINT"└──────────────────────
     ───────────────┘";
1860 PRINT"┌──────────────────────
     ───────────────┐";
1870 FOR A=0 TO 15
1880 PRINT"|
                    |";
1890 NEXT
1900 PRINT"└──────────────────────
     ───────────────┘";
1910 PRINT"[HOME]"
1920 PRINT"[5 DOWN][3 RIGHT]F1)PRO
     GRAMMAZIONE"
1930 PRINT"[DOWN][3 RIGHT]F3)ESECU
     ZIONE"
1940 PRINT"[DOWN][3 RIGHT]F5)VERIF
     ICA"
1950 PRINT"[DOWN][3 RIGHT]F7)CARIC
```

```
     AMENTO"
1960 PRINT"[DOWN][3 RIGHT]F2)SALVA
     TAGGIO"
1970 PRINT"[DOWN][3 RIGHT]F4)FINE"
1980 PRINT"[DOWN][3 RIGHT]F6)RANDO
     M"
1990 POKE 53269,1:POKE 53277,1:POK
     E 53271,1
2000 POKE 53248,255:POKE 53249,120
2010 POKE 53287,1:FOR A=1 TO 50:NE
     XT
2020 GET D$:IF D$="" THEN 2050
2030 IF D$<CHR$(133) OR D$>CHR$(13
     9) THEN 2050
2040 RETURN
2050 POKE 53287,0:FOR A=1 TO 50:NE
     XT:GOTO 2010
2060 :
2070 REM ******************
2080 REM *                *
2090 REM *     INPUT      *
2100 REM *                *
2110 REM ******************
2120 :
2130 C=1:B=1:Z=0:TM$=""
2140 PRINT"[HOME][BIANCO][2 DOWN][
     5 RIGHT]"PASSO;"    "
2150 PRINT"[HOME][BIANCO][5 DOWN][
     6 RIGHT]    "
2160 PRINT TAB(33)"[5 DOWN]       "
2170 N1=13:GOSUB 3510:PRINT"  CONF
     ERMI ?  SI<RETURN>     NO<SPA
     CE>  "
2180 FOR A=0 TO 7
2190 POKE 53269,B+Z
2200 POKE 53286+C,1:FOR W=1 TO 10:
     NEXTW
2210 GET K$:IF K$="" THEN 2240
2220 IF K$=CHR$(13) THEN Z=Z+B:GOT
     O 2260
2230 IF K$=CHR$(32) THEN 2260
2240 POKE 53286+C,0:FOR J=0 TO 10:
     NEXTJ
2250 GOTO 2190
2260 B=B*2:C=C+1:NEXTA
2270 POKE 53269,Z
2280 PRINT"[HOME][BIANCO][5 DOWN][
     6 RIGHT]"Z;"    "
2290 :
2300 REM ***************
2310 REM *    TIME     *
2320 REM ***************
2330 :
2340 KL=1
2350 TM$="":KL=1:REM  AZZERAMENTO
     TIME
2360 PRINT"[HOME]"
2370 PRINT TAB(34)"[7 DOWN]"C1$
2380 IF KL>6 THEN 2490
2390 GET YY$:IF YY$="" THEN 2460
2400 IF YY$=CHR$(13) AND (KL=1) TH
     EN TM$="10":GOTO 2490
2410 IF YY$=CHR$(13) THEN  2490
2420 IF ASC(YY$)<48 OR ASC(YY$)>57
      THEN 2460
2430 TM$=TM$+YY$:KL=KL+1
2440 PRINT"[HOME]":PRINT"[8 DOWN]"
2450 PRINT TAB(40-KL)TM$
2460 PRINT"[HOME]"
2470 PRINT TAB(34)"[7 DOWN]"CC$
2480 GOTO 2360
2490 PRINT"[HOME]"
2500 PRINT TAB(34)"[7 DOWN]"CC$
2510 N1=13:GOSUB 3510:PRINT"  CONF
     ERMI ?  SI<RETURN>     NO<SPA
     CE>  "
2520 N1=15:GOSUB 3510:PRINT"
          <←> PER USCIRE  "
2530 GET KK$:IF KK$="" THEN 2530
2540 IF KK$=CHR$(13) THEN 2590
2550 IF KK$=CHR$(32) THEN 2630
2560 IF KK$="←" THEN 2590
2570 GOTO 2530
2580 GOTO 2130
2590 PR(PA,0)=Z:PR(PA,1)=VAL(TM$)
2600 PR(PA,0)=Z:PR(PA,1)=VAL(TM$)
2610 PASSO=PA+1
2620 IF KK$="←" THEN RETURN
2630 GOTO 2130
2640 :
2650 REM ****************
2660 REM *              *
2670 REM *   VERIFICA   *
2680 REM *              *
2690 REM ****************
2700 :
2710 PA=0
2720 PRINT"[BIANCO] VERIFICA";"[UP
     ]"
2730 N1=13:GOSUB 3510:PRINT"[BIANC
     O]   <+>AVANTI <->INDIETRO <
     E>EDITA "
```

```
2740 PRINT TAB(12)"[2 DOWN]<←> PER
     USCIRE    "
2750 POKE 53269,PR(PA,0)
2760 PRINT"[HOME][BIANCO][2 DOWN][
     5 RIGHT]"PASSO;"[LEFT]        "
2770 PRINT"[HOME][BIANCO][5 DOWN][
     6 RIGHT]"PR(PA,0);"[LEFT]     "
2780 PRINT TAB(33)"[5 DOWN]        "
2790 PRINT TAB(33)"[2 UP]"MID$(STR
     $(PR(PA,1)),2,7)
2800 GET  D$:IF  D$="" THEN  2800
2810 IF  D$="-"  AND  PA>0  THEN
     PA=PA-1
2820 IF  D$="+"  AND  PA<1000  THE
     N  PA=PA+1
2830 IF D$="E" THEN GOSUB 2130:GOS
     UB 2870
2840 IF  D$="←"  THEN POKE 650,0:R
     ETURN
2850 FOR I=1  TO  10 :NEXTI
2860 GOTO 2750
2870 N1=13:GOSUB 3510:PRINT"     <+
     >AVANTI <->INDIETRO <E>EDITA
        "
2880 PRINT TAB(12)"[DOWN]<←> PER U.
     SCIRE"
2890 RETURN
2900 :
2910 REM *******************
2920 REM *                 *
2930 REM *     RANDOM      *
2940 REM *                 *
2950 REM *******************
2960 :
2970 POKE 53269,0:POKE 53280,12:PO
     KE 53281,12
2980 PRINT"[CLEAR][BIANCO]"
2990 PRINT"DURATA MINIMA CICLO":PR
     INT
3000 PRINT"DURATA MASSIMA CICLO"
3010 PRINT"[BLEU]┌──────────┬─────
     ┐
3020 FOR V=0 TO 6
3030 PRINT"|LAMPADA    |     %|
3040 PRINT"├──────────┼─────┤
3050 NEXT
3060 PRINT"|LAMPADA    |     %|
3070 PRINT"└──────────┴─────┘
3080 PRINT"[HOME]"
3090 PRINT TAB(20):LM=6:GOSUB 3280
     :MIN=S
3100 PRINT:PRINT TAB(21):GOSUB 328
     0:MAX=S
3110 FOR I=0 TO 7
3120 PRINT:PRINT TAB(8)I;
3130 PRINT TAB(12):LM=3:GQSUB 3280
     :PA(I)=S
3140 NEXT
3150 PRINT"[DOWN]QUANTI CICLI ? ";
     :LM=6:GOSUB 3280:CI=S
3160 FOR I=0 TO CI-1
3170 PR(I,1)=INT(RND(1)*(MAX-MIN)+
     MIN)
3180 T1=0
3190 PRINT"[HOME]":N1=10:GOSUB 351
     0:PRINT TAB(22);
3200 PRINT"CICLO NUMERO";I
3210 FOR J=0 TO 7
3220 IF (RND(1)*100<PA(J)) THEN T1
     =T1+2↑J
3230 NEXT
3240 PR(I,0)=T1
3250 NEXT
3260 PR(I,1)=0
3261 RETURN
3270 :
3280 REM ****************
3290 REM *              *
3300 REM * INPUT NUMERI *
3310 REM *              *
3320 REM ****************
3330 :
3340 S$=""
3350 PRINT"[BIANCO][RVOFF]>[LEFT]"
     ;
3360 GET L$:IF L$="" THEN 3410
3370 IF L$=CHR$(13) THEN 3420
3380 IF L$<CHR$(48) OR L$>CHR$(57)
      THEN 3360
3390 S$=S$+L$:PRINTL$;
3400 IF LEN(S$)=LM THEN 3420
3410 PRINT"[RVS]>[LEFT]";:GOTO 335
     0
3420 PRINT" ":S=VAL(S$)
3430 RETURN
3440 :
3450 REM *****************
3460 REM *               *
3470 REM * POS.VERTICALE *
3480 REM *               *
3490 REM *****************
3500 :
```

```
3510 PRINT"[HOME]"
3520 FOR BH=1 TO N1:PRINT:NEXT:RET
     URN
3530 :
3540 REM ***************
3550 REM *             *
3560 REM * SALVATAGGIO *
3570 REM *             *
3580 REM ***************
3590 :
3600 PRINT"[CLEAR]":POKE 53280,12:
     POKE 53281,12
3610 N1=10:GOSUB 3510
3620 INPUT "  NOME DEL FILE";NF$
3630 OPEN 15,8,15
3640 PRINT#15,"S:"NF$""
3650 CLOSE 15
3660 OPEN 2,8,2,"0:"+NF$+",S,W"
3670 J=0
3680 PRINT#2,PR(J,0)CHR$(13)PR(J,1
     )CHR$(13);
3690 J=J+1
3700 IF PR(J,1)<>0 THEN 3680
3710 CLOSE 2:RETURN
3720 :
3730 REM ***************
3740 REM *             *
3750 REM * CARICAMENTO *
3760 REM *             *
3770 REM ***************
3780 :
3790 POKE 53269,0
3800 PRINT"[CLEAR]":POKE 53280,12:
     POKE 53281,12
3810 N1=10:GOSUB 3510
3820 INPUT "  NOME DEL FILE";NF$
3830 OPEN 2,8,2,"0:"+NF$+",S,R"
3840 II=0
3850 INPUT#2,PR(II,0),PR(II,1)
3860 IF PR(II,0)<>0 THEN II=II+1:G
     OTO 3850
3870 CLOSE 2
3880 RETURN
3890 :
3900 REM **************
3910 REM *            *
3920 REM * ESECUZIONE *
3930 REM *            *
3940 REM **************
3950 :
3960 POKE  56579,255:REM  SETTA IN
     "OUT"
3970 PRINT" ";"[BIANCO]ESECUZIONE"
3980 PRINT TAB(13)"[4 DOWN]<←> PER
     USCIRE"
3990 I=0
4000 TEMPO=INT(PR(I,1)*TX)
4010 IF  TE=0  THEN  3990
4012 PRINT"[HOME][2 DOWN][6 RIGHT]
     "
4020 PRINT"[HOME][BIANCO][2 DOWN][
     5 RIGHT]"I;"     "
4030 PRINT"[HOME][BIANCO][5 DOWN][
     6 RIGHT]";"      ";"[4 LEFT]";P
     R(I,0)
4040 PRINT TAB(33)"[5 DOWN]        "
4050 PRINT TAB(33)"[2 UP]"MID$(STR
     $(PR(I,1)),2,7)
4060 POKE  56577,PR(I,0)
4070 POKE 53269,PR(I,0)
4080 FINE=TI+TE-11
4090 IF  TI<FINE  THEN  4090
4100 I=I+1
4110 IF  I=PS+1  THEN  GOTO  3990
4120 GET  D$:IF  D$="←" THEN  RETU
     RN
4130 GOTO  4000
4140 RETURN
4150 :
4160 REM **********
4170 REM *        *
4180 REM *  FINE  *
4190 REM *        *
4200 REM **********
4210 :
4220 POKE 53269,0:PRINT"[CLEAR]":N
     1=11:GOSUB 3510
4230 PRINT TAB(12)"ARRIVEDERCI !!"
4240 END
4250 DATA 0,124,0,1,131,0,3,0,192,
     6
4260 DATA 0,96,12,0,48,24,68,24,25
     ,171
4270 DATA 152,24,145,24,24,129,24,
     12,66,48
4280 DATA 6,36,96,3,36,192,3,255,1
     28,3
4290 DATA 255,128,1,1,0,3,255,128,
     1,1
4300 DATA 0,3,255,128,1,69,0,0,68,
     0
4310 DATA 0,56,0
```

BED

BINARIO * ESADECIMALE * DECIMALE

di **Eugenio Coppari**

Le civiltà che si sono susseguite dall'antichità sino ai nostri giorni hanno adottato svariati metodi di rappresentazione numerica.

La scelta di una rappresentazione rispetto ad un'altra è originata dalle particolari esigenze che debbono essere soddisfatte nel corso del suo utilizzo.

Le prime popolazioni terrestri erano solite rappresentare graficamente, un certo numero di oggetti, mediante dei segni incisi sul legno o sulla pietra.

Esempi più recenti testimoniano il largo sviluppo di rappresentazioni numeriche diverse da quella che usiamo attualmente con maggiore frequenza: la base 10.

Nella scelta di una determinata opportunità rispetto ad un'altra, incide soprattutto la sua capacità di adattamento a ciò che si utilizza per quantificare un certo numero di elementi: probabilmente oggi saremmo maggiormente portati ad usare la base 20 se piuttosto di 10 fossimo stati in possesso di un numero doppio di dita.

Questo discorso ha una particolare validità nei confronti di una disciplina come l'informatica: analizziamone i motivi.

Ogni calcolatore, durante il suo funzionamento, gestisce una notevole quantità di informazioni.

La rappresentazione interna di queste informazioni avviene in formato binario a causa degli elevati costi che imporrebbe una scelta non ristretta a due soli stati: zero oppure uno.

Le più piccole informazioni rappresentabili sono i bit (detti anche digit binari), essi vengono solitamente aggregati a gruppi di otto che vengono denominati byte.

Ritengo opportuno, per coloro che non se ne ricordassero, richiamare la definizione di base numerica.

Prendiamo come riferimento la base 10 che è quella normalmente usata nei nostri calcoli quotidiani.

I simboli grafici fondamentali utilizzati nel sistema decimale sono i seguenti: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

In un sistema di questo tipo, il valore di un simbolo grafico fondamentale è funzione della sua posizione all'interno del numero.

Il metodo di conta in tale sistema è il seguente:

a) tutti i simboli devono essere considerati sequenzialmente da 1 sino a 9

b) se la quantità da contare è superiore a 9, tutti i simboli devono essere considerati nuovamente a partire da 0 e deve essere effettuato il riporto, cioè si deve sommare 1 alla cifra immediatamente alla sinistra (quella delle decine)

c) se la cifra delle decine raggiunge il valore 9, il procedimento ora descritto deve essere ripetuto ed il riporto va effettuato sulla cifra immediatamente alla sinistra (quella delle centinaia)

d) il procedimento va ripetuto senza alcuna limitazione sul numero

delle cifre interessate.
In conseguenza di ciò che è stato detto il numero 8352 è la rappresentazione simbolica di: 8 per 10 elevato al cubo più 3 per 10 elevato al quadrato più 5 per 10 elevato alla prima più 2 per 10 elevato allo zero.

8 × 10 elevato a 3	= 8 ×	1000 +
3 × 10 elevato a 2	= 3 ×	100 +
5 × 10 elevato a 1	= 5 ×	10 +
2 × 10 elevato a 0	= 2 ×	1 +
		8352

Per quanto concerne la numerazione binaria il discorso è del tutto analogo a quello precedente, tenendo però conto del fatto, che le cifre disponibili sono due, 0 e 1, e quindi il riporto dovrà essere effettuato quando si giunge a 1.
Discorsi del tutto analoghi riguardano il sistema esadecimale che si avvarrà, in più rispetto a quello decimale, delle cifre A, B, C, D, E, F.
Passiamo ora all'analisi del listato che consente sei diversi tipi di conversioni tra i tre sistemi in esame: decimale, binario e esadecimale.
Nelle linee comprese tra 150 e 230 viene creata la mascherina iniziale del programma.

Le righe da 270 sino a 430 costituiscono la routine di input controllato: essa consente di impedire l'entrata di elementi incompatibili con il sistema oggetto di INPUT.
Se ad esempio vogliamo effettuare una conversione da esadecimale nei due altri sistemi di numerazione, non verranno accettati simboli diversi da quelli appartenenti al seguente insieme:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

La locazione di memoria 204 posta a 0 consente di ottenere il cursore bistabile (lampeggiante). Esso si troverà nella posizione di memoria immediatamente seguente a quella dell'ultima PRINT.
La routine di input controllato viene creata avvalendosi dei codici ASCII dei caratteri e delle istruzioni stringa disponibili sul Commodore 64.
Il range numerico consentito per le conversioni di ogni sistema è il seguente:

decimale	0 - 999999999
esadecimale	0 - FFFFFF
binario	0 - 11111111111111111111111111111111

Valori superiori a quelli citati non verranno presi in considerazione, questa limitazione è dovuta ai problemi di precisione che coinvolgono il microprocessore del computer.
Voglio trarre spunto da questo problema per annunciare che in un prossimo numero verrà affrontato il discorso della N precisione, esso risulterà particolarmente interessante per tutti coloro che hanno esigenze di maggiore precisione (grandezza) sul proprio computer.
Le linee da 460 a 525 consentono le conversioni da decimale in esadecimale e da decimale in binario. Queste conversioni avvengono, rispettivamente, tramite delle successive divisioni per 16 e per 2.
Le righe da 555 a 600 permettono i cambiamenti di base da esadecimale in decimale e da esadecimale in binario.
La seconda conversione viene effettuata passando attraverso il passaggio intermedio del decimale.
Le linee da 630 a 645 effettuano la conversione binario in decimale e da binario in esadecimale.
La seconda delle due viene anch'essa effettuata passando attraverso lo stadio decimale.

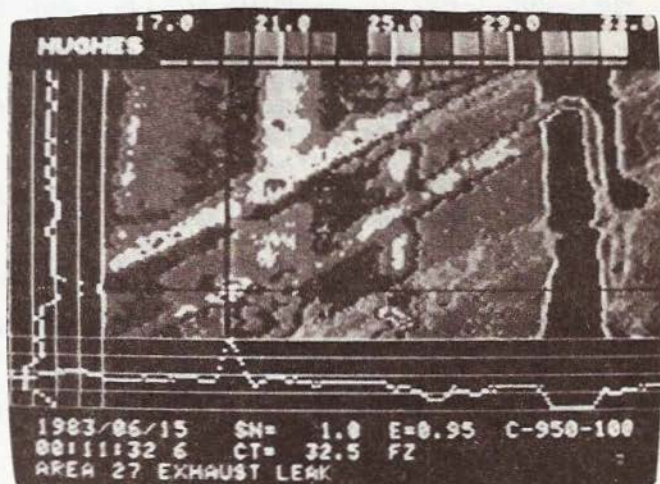
```
100 REM ***************************
105 REM *                         *
110 REM * CONVERSIONI NUMERICHE   *
115 REM *                         *
120 REM *   DI  EUGENIO COPPARI   *
125 REM *                         *
130 REM * VIA VIGANO' 4 (MILANO)  *
135 REM *                         *
140 REM *      TEL:659.11.52      *
145 REM *                         *
150 REM ***************************
155 PRINTCHR$(147)
160 Q$="ABCDEF"
165 POKE 53280,0:POKE 53281,0
170 PRINT TAB(9)"┌──────────────
    ──┐"
175 PRINT TAB(9)"| ** CONVERSIONI
    ** |"
180 PRINT TAB(9)"└──────────────
    ──┘"
185 LK=LK+1:SD$(LK)="DECIMALE:"
190 LK=LK+1:SD$(LK)="ESADECIMALE:"
195 LK=LK+1:SD$(LK)="BINARIO:"
200 FOR TY=1 TO 3
205 PRINT"[3 DOWN]┌──────────────────
    ──────────────────┐"
210 PRINT"[UP]|
                 |"
215 PRINT"[UP]└──────────────────────
    ──────────────────┘"
220 PRINT TAB(( SGN(TY)-1)*100+40)
    ;"[4 UP][RIGHT]";SD$(TY)
225 NEXT
230 PRINT"[HOME][3 DOWN]"; TAB(210
    ):POKE 204,0:R$="D"
235 REM ***************************
240 REM *                         *
245 REM *    ROUTINE DI INPUT     *
250 REM *                         *
255 REM *      CONTROLLATO        *
```

```
260 REM *                          *
265 REM ****************************
270 GET B$:IF B$="" THEN 270
275 IF B$=CHR$(13) AND A$<>"" THEN
     POKE 204,1:PRINT" ";:GOTO 375
280 IF LEN(A$)=9 THEN 270
285 IF B$="[DOWN]" AND A$="" THEN
    POKE 204,1:PRINT" ":GOTO 300
290 IF (ASC(B$)<48 OR ASC(B$)>57)
    THEN 270
295 A$=A$+B$:PRINTB$;:GOTO 270
300 PRINT TAB(173);:POKE 204,0:R$=
    "H"
305 GET B$:IF B$="" THEN 305
310 IF B$=CHR$(13) AND A$<>"" THEN
     POKE 204,1:PRINT" ";:GOTO 375
315 IF LEN(A$)=7 THEN 305
320 IF B$="[DOWN]" AND A$="" THEN
    POKE 204,1:PRINT" ":GOTO 340
325 IF B$="[UP]" AND A$="" THEN PO
    KE 204,1:PRINT" ":GOTO 230
330 IF ((ASC(B$)<48 OR ASC(B$)>57)
     AND (ASC(B$)<65 OR ASC(B$)>70
    )) THEN 305
335 A$=A$+B$:PRINTB$;:GOTO 305
340 PRINT TAB(169);:POKE 204,0:R$=
    "B"
345 GET B$:IF B$="" THEN 345
350 IF B$=CHR$(13) AND A$<>"" THEN
     POKE 204,1:PRINT" ";:GOTO 375
355 IF LEN(A$)=29 THEN 345
360 IF B$="[UP]" AND A$="" THEN PO
    KE 204,1:PRINT" ";"[HOME][5 DO
    WN]"; TAB(210):GOTO 300
365 IF (B$<>"0" AND B$<>"1") THEN
    345
370 A$=A$+B$:PRINTB$;:GOTO 345
375 IF R$<>"D" THEN 400
380 L=2:A=VAL(A$):GOSUB 460
385 CL$="":L=16:A=VAL(A$):GOSUB 46
    0
390 GET UZ$:IF UZ$="" THEN 390
395  RUN
400 IF R$<>"H" THEN 420
405 L=2:GOSUB 555:GOSUB 525
410 GET UZ$:IF UZ$="" THEN 410
415  RUN
420 IF R$="B" THEN L=16:GOSUB 630:
    GOSUB 525
425 GET UZ$:IF UZ$="" THEN 425
430  RUN
435 REM ****************************
440 REM *                          *
445 REM * CONVERSIONE: D/H - D/B *
450 REM *                          *
455 REM ****************************
460 Z=INT(A/(L↑K))
465 IF Z>(L-1) THEN K=K+1:GOTO 460
470 GOTO 480
475 K=K-1:Z=INT(A/(L↑K))
480 IF L=2 THEN 500
485 FOR YU=10 TO 15
490 IF Z=YU THEN CL$=CL$+MID$(Q$,Y
    U-9,1):GOTO 505
495 NEXTYU
500 CL$=CL$+RIGHT$(STR$(Z),1)
505 IF K=0 THEN 515
510 A=A-Z*(L↑K):GOTO 475
515 IF L=16 THEN PRINT"[HOME][13 D
    OWN][13 RIGHT][RVS]";CL$:RETUR
    N
520 IF L=2 THEN PRINT"[HOME][18 DO
    WN][9 RIGHT][RVS]";CL$:RETURN
525 PRINT"[HOME][8 DOWN][10 RIGHT]
    [RVS]";MID$(STR$(YR),2):RETURN
530 REM ****************************
535 REM *                          *
540 REM * CONVERSIONE: H/D - H/B *
545 REM *                          *
550 REM ****************************
555 I=LEN(A$)
560 FOR DW=1 TO I
565 FOR TU=10 TO 15
570 IF MID$(A$,DW,1)=MID$(Q$,TU-9,
    1) THEN A=A+TU*(16↑(I-DW)):GOT
    O 595
575 NEXTTU
580 GOTO  590
585 NEXTDW
590 A=A+VAL(MID$(A$,DW,1))*(16↑(I-
    DW))
595 IF DW=I THEN YR=A:GOTO 460
600 GOTO 585
605 REM ****************************
610 REM *                          *
615 REM * CONVERSIONE: B/D - B/H *
620 REM *                          *
625 REM ****************************
630 I=LEN(A$)
635 W=W+1:A=A+VAL(MID$(A$,W,1))*(2
    ↑(I-W))
640 IF W=I THEN YR=A:GOTO 460
645 GOTO 635
```

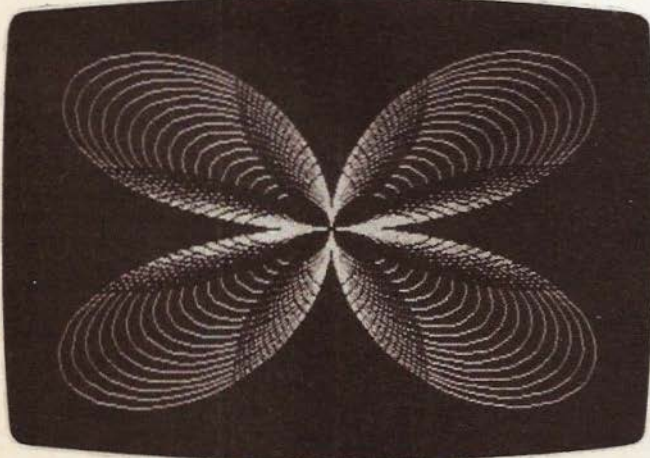
EQUAZIONE DELLA RETTA E SUA RAPPRESENTAZIONE GRAFICA

di **Eugenio Coppari**

L'utilizzo del calcolatore nella ricerca matematica non si limita semplicemente alla semplificazione e velocizzazione del calcolo numerico. Esistono in effetti dei programmi che sono in grado di manipolare e ridurre espressioni algebriche di notevole complessività. Il listato che vi proponiamo permette di calcolare l'equazione di una retta qualora siano state fornite al computer 2 coppie di coordinate.
Una operazione di questo genere richiede naturalmente una gestione da parte della macchina di dati alfanumerici.
Inoltre il programma vi fornirà l'angolo formato dalla retta con l'asse delle ascisse e la misura del segmento avente per estremi le 2 coppie di coordinate.
A conclusione di queste fasi saremo in grado di visualizzare i segmenti e le loro intersezioni in alta risoluzione tramite il modo bit-map.
Passiamo ora all'analisi del listato.
Le righe che vanno da 139 a 232 consentono la presentazione del programma mediante una mascherina iniziale, da 181 a 193 si fa ricorso ad alcune Poke dello schermo video e relative locali del colore.
Nelle righe da 256 a 448 viene effettuato il calcolo dell'angolo e della misura del segmento, vediamo nel dettaglio come avvengono queste operazioni.
Da 268 a 328 vengono visualizza-

te mediante un grafico le variabili che rappresentano gli elementi oggetto di calcolo nel programma (angolo e segmento).
A 334 è situata la formula che consente il calcolo dell'angolo Q, essa viene espressa tramite l'istruzione BASIC DEF FN che consente di definire con un breve nome un calcolo complesso tipo una funzione.
La riga 367 consente il calcolo della misura del segmento attraverso una applicazione del teorema di Pitagora alle coppie di coordinate iniziali.

La matrice FT (20) permette la memorizzazione delle coordinate introdotte dall'utente durante i calcoli successivi al primo che si è effettuato.
Le righe da 409 a 421 consentono il compattamento della stringa che rappresenta l'equazione della retta in esame.
Nella riga 431 viene evidenziato il numero massimo di calcoli consecutivi che permette il programma (totale 5).
Questa limitazione è dovuta alla particolare struttura di memoria che caratterizza il Commodore 64. Poiché il programma, le variabili e le matrici del BASIC vengono caricate in memoria, salvo controindicazioni iniziali, a partire dalla locazione decimale 2048 bisogna prestare particolare attenzione al fatto che uno di questi elementi non invada la pagina grafica che è stata locata a partire da 8192.
Per chi lo desiderasse è possibile togliere questa limitazione spostando l'area BASIC al di sopra della locazione 16192, in quanto la pagina grafica occupa 8k di memoria.

Anche in conseguenza di ciò che si è detto si raccomanda di eliminare le righe provviste di REM prima dell'esecuzione del programma.
Nelle righe da 469 a 511 viene stimato il tempo di lavoro dell'utente, esso può risultare interessante nel caso in cui si cerchi di calcolare autonomamente i risultati prima che li comunichi il computer. Infatti il tempo che viene visualizzato non è altro che il periodo che intercorre tra l'inizio del primo calcolo (momento di digitazione delle prime coordinate) e la fine dell'inputazione.
Da 535 a 538 è definita la routine in linguaggio macchina che consente una pulizia estremamente veloce della pagina grafica.
Essa viene allocata a partire dalla locazione decimale 49152 ($C000), cioè in quella area di 4K RAM che non è soggetta all'interprete BASIC.
Ora analizzeremo la parte probabilmente più interessante del listato: il calcolo dell'equazione.
Questo calcolo è compreso tra le righe 595 e 661 e si avvale delle funzioni di stringa più importanti che possiede il BASIC del Commodore 64.

Per calcolare l'equazione di una retta, date 2 coppie di coordinate, è fondamentale determinare i valori dei coefficenti A1, B1 e C1 dell'equazione generale di primo grado della retta A1X + B1Y + C1.
Questi elementi sono dati dalle seguenti relazioni: A1 è uguale alla differenza delle ordinate A1 = y2—y1), B1 è uguale alla differenza delle ascisse (B1 = x2-x1), infine C1 è dato dalla differenza tra il prodotto della seconda ascissa per la prima ordinata e quello della prima ascissa per la seconda ordinata (C1 = x2y1-x1y2).
Le righe da 595 a 601 eseguono i calcoli appena citati tenendo conto del fatto che, per convenzione, il primo coefficiente dell'equazione deve essere positivo.
Le righe da 604 sino a 622 valutano i valori dei coefficienti A1, B1 e C1 ed in base a questi si attiva quella parte di programma idonea per il calcolo della stringa che ci fornirà l'equazione.
Facciamo ora un esempio di quanto è stato affermato: immaginiamo di avere tutti e tre i valori di A1, B1 e C1 diversi da zero, in questo caso il programma si recherà alla riga 637 dove troverà la struttura confacente allo sviluppo di una stringa che rappresenti correttamente l'equazione che deve risultare.
I GOTO posti al termine di ogni riga da 628 sino 661 vengono valutati solamente se la condizione in esame nella linea è verificata, questo comportamento è tipico della istruzione IF...THEN... del BASIC.

Le righe da 682 sino a 742 permettono la costruzione in pagina grafica degli assi cartesiani orientati.
Nelle linee da 766 a 844 avviene la visualizzazione dei segmenti in pagina grafica nell'ordine in cui sono stati richiesti inizialmente dall'utente.
Poiché il Commodore 64 ha una risoluzione grafica di 320 per 200 pixel, punti, l'origine degli assi cartesiani è stata posta alle coordinate 159 per le ascisse e 99 per le ordinate.
A questo proposito ricordiamo che in pagina grafica le coordinate del computer hanno inizio nell'angolo in alto a sinistra.
Le righe da 772 a 781 traslano le coordinate reali date inizialmente dall'utente per adeguarle alla situazione esposta nella frase precedente.
La riga 783 impedisce al segmento che abbia le ascisse al di fuori del range consentito dal programma di essere disegnato (0 319 oppure -159 159 in valore utente), poiché altrimenti si verificherebbero, inevitabilmente, dei fenomeni di spezzatura del segmento.
Le linee da 790 sino a 796 definiscono le funzioni necessarie per il disegno di un punto della retta:
— DEF FNCH (Z) è la formula usata per determinare il numero di riga e di colonna del carattere che corrisponde alle coordinate del punto;
— DEF FNCO (Z) serve per calcolare il byte a cui appartiene il punto attualmente in esame (si parte per quanto concerne il calcolo dalla 8192);
— DEF FNBI (Z) serve per calcolare quale tra gli 8 bit disponibili sia quello corrispondente al nostro punto.
Le procedure che sono state ora elencate potrebbero apparire complesse, ma se le osserviamo più attentamente esse ricalcano l'impostazione suggerita dalla: "Guida di riferimento per il Commodore 64" a riguardo il lavoro in pagina grafica.
Nelle righe 802 e 805 il programma deve scegliere tra 2 possibilità che incideranno sul proporzionamento del segmento al fine di renderlo il più verosimigliante possibile ad uno reale infatti il programma esegue un rapporto fra la differenza in valore assoluto delle ascisse e le ordinate. L'iterazione in 841 avrà termine quando saranno esaurite le coordinate inserite dall'utente.
Questo programma si presta senz'altro a notevoli ampliamenti per quanto concerne gli enti geometrici che possono essere oggetto di rappresentazione.

```
100 REM ***************************
103 REM *                         *
106 REM * EQUAZIONE DELLA  RETTA *
109 REM *                         *
112 REM * E SUA RAPPRESENTAZIONE *
115 REM *                         *
118 REM *        GRAFICA .        *
121 REM *                         *
124 REM *  DI EUGENIO COPPARI     *
127 REM *                         *
130 REM *  TEL : 659.11.52 (MI)   *
133 REM *                         *
136 REM ***************************
139 PRINTCHR$(147);"[DOWN]"
142 FOR JK=1 TO 23
145 IF JK=12 THEN PRINT TAB(2);"___
    ____________________________
    ___":GOTO 154
148 IF JK=13 THEN PRINT:GOTO 154
151 PRINT"[BIANCO][19 RIGHT]| "
154 NEXT
157 REM ***************************
160 REM *                         *
163 REM * PRIMA DI FAR GIRARE IL *
166 REM * PROGRAMMA E'NECESSARIO *
169 REM * ELIMINARE LE RIGHE CON *
172 REM * TENENTI LE REM .        *
175 REM *                         *
178 REM ***************************
181 POKE 2003,116:POKE 56275,1
184 POKE 1042,233:POKE 55314,1
187 POKE 1043,223:POKE 55315,1
190 POKE 1542,223:POKE 55814,1
193 POKE 1582,105:POKE 55854,1:PRI
    NT"[HOME]"
196 FOR BJ=1 TO 5
199 IF BJ/2<>INT(BJ/2) THEN PRINT
    TAB(31-BJ);"/":GOTO 205
202 PRINT
205 NEXT:PRINT"[DOWN]";
208 FOR BJ=8 TO 18
211 PRINT TAB(32-BJ);"/"
214 NEXT:PRINT"[DOWN]";
217 FOR BJ=1 TO 5
220 IF BJ/2<>INT(BJ/2) THEN PRINT
    TAB(13-BJ);"/":GOTO 226
223 PRINT
226 NEXT
229 PRINT"[HOME]";"[RVS]
         [RVOFF]"
232 PRINT"[RVS] EQUAZIONE DELLA [R
    VOFF]":PRINT"[RVS]      RETTA
         [RVOFF]":PRINT"[RVS]
              [RVOFF]"
235 REM ***************************
238 REM *                         *
```

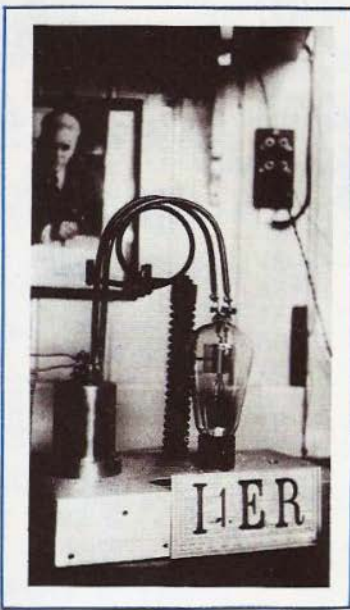
```
241 REM * CALCOLO DELL' ANGOLO    *
244 REM * E DELLA  MISURA  DEL    *
247 REM *        SEGMENTO .       *
250 REM *                         *
253 REM ***************************
256 FOR RZ=1 TO 2000:NEXT
259 TI$="000000"
262 DIM FT(20)
265 POKE 53280,1:POKE 53281,1
268 PRINT"[NERO]"
271 PRINT"[CLEAR]"
274 PRINT TAB(4)"Y":GOSUB 682
277 PRINT TAB(3)"↑"
280 FOR A=1 TO 10
283 PRINT TAB(3)"|"
286 NEXTA:GOSUB 682
289 PRINT TAB(4)"──────────>":GOSUB
    682
292 PRINT TAB(15)"X"
295 PRINT"[HOME]"
298 PRINT TAB(12)"[3 DOWN][RVS][BL
    EU]O":PRINT TAB(14)"P2":GOSUB
    682
301 PRINT TAB(5)"[5 DOWN][RVS][BLE
    U]O[RVOFF][ROSSO]......":PRINT
     TAB(7)"[BLEU]P1[NERO]":GOSUB
    682
304 PRINT"[HOME][4 DOWN]"
307 FOR A=11 TO 6 STEP -1
310 PRINT TAB(A)"[ROSSO]/[NERO]"
313 NEXTA:GOSUB 682
316 PRINT"[HOME][7 DOWN][8 RIGHT][
    ROSSO]W":GOSUB 682
319 PRINT"[HOME][9 DOWN][8 RIGHT]\
    Q"
322 PRINT"[HOME][10 DOWN][8 RIGHT]
    |"
325 PRINT"[HOME][11 DOWN][8 RIGHT]
    /":GOSUB 682
328 PRINT"[2 DOWN]"
331 IF ID=1 THEN 394
334 DEF FNAC(X)=(-ATN(X/SQR(-X*X+1
    ))+π/2)*180/π
337 X=0:Y=0:A=0:B=0
340 INPUT "[2 DOWN][ROSSO]P1  X,Y[
    NERO] ";X,Y:X1=X:Y1=Y
343 GOSUB 862:FT(EW)=X1
346 GOSUB 862:FT(EW)=Y1
349 INPUT "[ROSSO]P2  X,Y[NERO] ";
    A,B:X2=A:Y2=B
352 GOSUB 862:FT(EW)=X2
355 GOSUB 862:FT(EW)=Y2
358 ZC=ZC+1
361 GOSUB 595
364 C=A-X:D=B-Y
367 E=SQR(C*C+D*D)
370 IF C=0 AND E=0 THEN F=0:GOTO 3
    79
373 IF D=0 THEN F=0:GOTO 379
376 F=FNAC(C/E)
379 IF 0>D THEN F=-F
382 F=F+ SGN(F)*.0005
385 F=INT(F*1000+.5)/1000
388 E=INT(E*1000+.5)/1000
391 ID=1:GOTO 268
394 PRINT"[RVS][ROSSO]P1 X[NERO][R
    VOFF]";X1,"[RVS][ROSSO]Y[RVOFF
    ][NERO]";Y1
397 PRINT"[RVS][ROSSO]P2 X[NERO][R
    VOFF]";X2,"[RVS][ROSSO]Y[RVOFF
    ][NERO]";Y2
400 PRINT"[RVS][ROSSO]W[NERO][RVOF
    F]",E
403 IF F<0 THEN F=F+180
406 PRINT"[RVS][ROSSO]Q[NERO][RVOF
    F]",F
409 E=LEN(O$)
412 FOR RI=1 TO E
415 VV$=MID$(O$,RI,1)
418 IF VV$=" " THEN O$=LEFT$(O$,RI
    -1)+RIGHT$(O$,E-(RI)):GOTO 409
421 NEXT
424 PRINT"[RVS][ROSSO]EQUAZIONE[NE
    RO][RVOFF] ";O$
427 PRINT"[DOWN][RVS][GIALLO]ANCOR
    A UN CALCOLO  S/N[DOWN]"
430 PRINT"[UP]            [NERO]O[LEFT
    ]";
431 IF ZC=5 THEN Q$="N":FOR VZ=1 T
    O 5000:NEXTVZ:GOTO 445
433 GET Q$:IF Q$="" THEN 433
436 PRINTQ$;"[BIANCO]":FOR QQ=1 TO
     1400:NEXT
439 PRINT"[NERO]"
442 IF Q$="S" THEN ID=0:GOTO 265
445 IF Q$="N" THEN 469
448 PRINT"[2 UP]":GOTO 430
451 REM ***************************
454 REM *                         *
457 REM *  TEMPO  DI LAVORO COL   *
460 REM *        COMPUTER .       *
463 REM *                         *
466 REM ***************************
469 PRINT"[CLEAR]"
```

```
472 PRINT"[7 DOWN]        [VERDE][RV
    S]
    [RVOFF]"
475 PRINT"       [VERDE][RVS]
    TEMPO  DI LAVORO       [RVOFF]"
478 PRINT"       [VERDE][RVS]
                           [RVOFF]"
481 H1$=LEFT$(TI$,2)
484 M1$=MID$(TI$,3,2)
487 S1$=RIGHT$(TI$,2)
490 H1=VAL(H1$):M1=VAL(M1$):S1=VAL
    (S1$)
493 PRINT"        ";"[NERO][RVS]
                           [RVOFF
    ][VERDE]"
496 PRINT"        ";"[NERO][RVS] [RV
    OFF][VERDE]       ";"  [NERO][R
    VS] [RVOFF][VERDE]     ";"     [
    NERO][RVS] [RVOFF][VERDE]
    ";"   [NERO][RVS] [RVOFF][VERD
    E]"
499 PRINT"        ";"[NERO][RVS] [RV
    OFF][VERDE]       ";"  [NERO][R
    VS] [RVOFF][VERDE]     ";"     [
    NERO][RVS] [RVOFF][VERDE]
    ";"   [NERO][RVS] [RVOFF][VERD
    E]"
502 PRINT TAB(8);"[UP]H =";H1; TAB
    (17);"M =";M1; TAB(26)"S =";S1
505 PRINT"        ";"[NERO][RVS] [RV
    OFF][VERDE]       ";"  [NERO][R
    VS] [RVOFF][VERDE]     ";"     [
    NERO][RVS] [RVOFF][VERDE]
    ";"   [NERO][RVS] [RVOFF][VERD
    E]"
508 PRINT"        ";"[NERO][RVS]
                           [RVOFF
    ][VERDE]"
511 FOR WT=1 TO 5000:NEXT
514 REM **************************
517 REM *                        *
520 REM * ROUTINE IN L.M. PER LA *
523 REM * PULITURA DELLA  PAGINA *
526 REM *        GRAFICA .       *
529 REM *                        *
532 REM **************************
535 FOR K=49152 TO 49222:READ P:PO
    KE K,P:NEXT
538 PRINT"[CLEAR]":BA=8192:SYS4915
    2
541 GOTO  685
544 DATA 173,24,208,9,8,141,24,208
547 DATA  173,17,208,9,32,141,17,2
    08
550 DATA  169,0,133,251,169,32,133
    ,252
553 DATA  160,0,169,0,145,251,200,
    192
556 DATA  0,208,249,230,252,169,64
    ,197
559 DATA  252,208,239,169,0,133,25
    1,169
562 DATA  4,133,252,160,0,169,3,14
    5
565 DATA  251,200,192,0,208,249,23
    0,252
568 DATA  169,8,197,252,208,239,96
571 REM **************************
574 REM *                        *
577 REM * LAVORO CON LE STRINGHE *
580 REM * DI  CARATTERI  PER  IL *
583 REM * CALCOLO DELL' EQUAZIO- *
586 REM * NE DELLA RETTA .       *
589 REM *                        *
592 REM **************************
595 IF B>Y THEN A1=B-Y:B1=X-A:C1=A
    *Y-X*B
598 IF B<Y THEN A1=Y-B:B1=A-X:C1=X
    *B-A*Y
601 IF B=Y THEN A1=0:B1=X-A:C1=A*Y
    -X*B
604 IF A1=0 AND B1<>0 AND C1<>0 TH
    EN 628
607 IF A1=0 AND B1<>0 AND C1=0 THE
    N 634
610 IF A1<>0 AND B1<>0 AND C1<>0 T
    HEN 637
613 IF A1<>0 AND B1<>0 AND C1=0 TH
    EN 649
616 IF A1<>0 AND B1=0 AND C1<>0 TH
    EN 655
619 IF A1<>0 AND B1=0 AND C1=0 THE
    N 661
622 O$="":RETURN
625 RETURN
628 IF C1<0 THEN O$=STR$(B1)+"Y"+S
    TR$(C1):GOTO 625
631 IF C1>0 THEN O$=STR$(B1)+"Y"+"
    +"+STR$(C1):GOTO 625
634 O$=STR$(B1)+"Y":GOTO 625
637 IF B1<0 AND C1>0 THEN O$=STR$(
    A1)+"X"+STR$(B1)+"Y"+"+"+STR$(
    C1):GOTO 625
640 IF B1>0 AND C1<0 THEN O$=STR$(
```

```
    A1)+"X"+"+"+STR$(B1)+"Y"+STR$(
    C1):GOTO 625
643 IF B1<0 AND C1<0 THEN O$=STR$(
    A1)+"X"+STR$(B1)+"Y"+STR$(C1):
    GOTO 625
646 IF B1>0 AND C1>0 THEN O$=STR$(
    A1)+"X"+"+"+STR$(B1)+"Y"+"+"+S
    TR$(C1):GOTO 625
649 IF B1>0 THEN O$=STR$(A1)+"X"+"
    +"+STR$(B1)+"Y":GOTO 625
652 IF B1<0 THEN O$=STR$(A1)+"X"+S
    TR$(B1)+"Y":GOTO 625
655 IF C1>0 THEN O$=STR$(A1)+"X"+"
    +"+STR$(C1):GOTO 625
658 IF C1<0 THEN O$=STR$(A1)+"X"+S
    TR$(C1):GOTO 625
661 O$=STR$(A1)+"X":GOTO 625
664 REM **************************
667 REM *                        *
670 REM *  VISUALIZZAZIONE DEGLI *
671 REM *        ASSI            *
673 REM *  CARTESIANI ORIENTATI. *
676 REM *                        *
679 REM **************************
682 FOR K8=1 TO 130:NEXT:RETURN
685 FOR U7=0 TO 39
688 POKE 12035+U7*8,255
691 NEXT
694 FOR  ET=1 TO 14
697 READ  RI:READ UH
700 POKE RI,UH
703 NEXT
706 FOR W6=0 TO 24
709 FOR EI=0 TO 7
712 AP=16039-320*W6-EI
715 IF AP=12195 THEN POKE AP,255:G
    OTO 721
718 POKE AP,128
721 NEXTEI
724 NEXT
727 FOR  PY=0 TO 7
730 READ UT
733 POKE  8352+PY,UT
736 READ UT
739 POKE  8344+PY,UT
742 NEXT
745 REM **************************
748 REM *                        *
751 REM *  RAPPRESENTAZIONE GRA- *
754 REM *  FICA DEL SEGMENTO NEL *
757 REM *  MODO BIT - MAP .      *
760 REM *                        *
763 REM **************************
766 EW=0
769 UL=UL+1
772 GOSUB 862:X1=FT(EW)+159
775 GOSUB 862:Y1=99-FT(EW)
778 GOSUB 862:X2=FT(EW)+159
781 GOSUB 862:Y2=99-FT(EW)
783 IF (X1>319 OR X1<0) OR (X2>319
     OR X2<0) THEN 841
784 GOSUB 790
787 GOTO 802
790 DEF FNCO(Z)=8192+320*FNCH(YR)+
    8*FNCH(XR)+YR-8*FNCH(YR)
793 DEF FNCH(Z)=INT(Z/8)
796 DEF FNBI(Z)=7+8*FNCH(Z)-Z
799 RETURN
802 IF ABS(X1-X2)>=ABS(Y1-Y2) THEN
     GOSUB 808
805 IF ABS(X1-X2)<ABS(Y1-Y2) THEN
    GOSUB 826
808 S=0:IF X1-X2<>0 THEN S=(Y1-Y2)
    /(X1-X2)
811 FOR XR=INT(X1+.5) TO INT(X2+.5
    ) STEP  SGN(X2-X1)
814 YR=INT((XR-X1)*S+.5+Y1)
817 P=FNCO(0):IF P>8191 AND P<1619
    2 THEN POKE P,PEEK(P) OR 2↑FNB
    I(XR)
820 NEXTXR
823 GOTO 841
826 S=0:IF Y1-Y2<>0 THEN S=(X1-X2)
    /(Y1-Y2)
829 FOR YR=INT((Y1)+.5) TO INT(Y2+
    .5) STEP  SGN(Y2-Y1)
832 XR=INT((YR-Y1)*S+X1)
835 P=FNCO(0):IF P>8191 AND P<1619
    2 THEN POKE P,PEEK(P) OR 2↑FNB
    I(XR)
838 NEXTYR
841 IF UL<ZC THEN 769
844 GOTO 844
847 DATA  12348,254,12346,254,1234
    9,252,12345,252,12350,248,1234
    4,248
850 DATA  12351,240,12028,128,1266
    6,128,12029,192,12665,192,1203
    0,224
853 DATA  12664,224,12031,240
856 DATA  128,1,192,3,224,7,240,15
859 DATA  248,31,252,63,254,127,25
    5,255
862 EW=EW+1:RETURN
```

IL COMPATTORE PER TUTTI I TIPI DI COMMODORE

di **Giancarlo De Cobelli**

La continua diffusione del Commodore ha portato il mercato di software disponibile per questo computer a livelli veramente ottimi.
Oltre tutto la rapida nascita di innumerevoli Club Commodore ha permesso alla maggior parte di utilizzatori di possedere una fornita libreria di programmi di svariato genere. Lo scopo di questo articolo è quello di divulgare, anche per utilizzatori meno attenti ai vari movimenti del 'mercato' dei programmi, una utility che io ritengo possa mancare solo a chi si accontenta di usare il computer per videogiocare con gli innumerevoli giochi disponibili per il Commodore. Il listato pubblicato in queste pagine è una modifica del programma originale apparso su una rivista, non ricordo se americana, dell'aprile dello scorso anno. Il programma è stato tradotto in italiano e modificato per poter permettere la scelta del drive in input e di output ed è stata aggiunta una routine di fine programma.
Il compattore, così si chiama questa utility, già dal nome lascia capire quale sia il suo utilizzo. Molte volte vi sarà senz'altro capitato di dover diminuire (per i più svariati motivi) lo spazio di memoria occupato da un programma. In quel momento vi siete visti costretti a passare riga per riga il programma per eliminare tutti i commenti ('REM') e gli spazi tra una lettera e l'altra. Ora basta! Digitando questo sensazionale listato vedrete i vostri programmi compattati in un baleno.
A parte gli scherzi, questa utility permette di velocizzare in maniera notevole il compattamento di un programma. Un consiglio (se siete in possesso di un compilatore BASIC) è quello di compilare il Compattatore in modo tale che il risultato possa rendere più veloce ogni esecuzione futura del programma.
Il Compattatore molte volte si rivela utile ma anche in questi casi in cui risulti necessario eliminare alcune protezioni basate sull'utilizzo delle REM, infatti elimina tutte le REM presenti nel programma.

Funzionamento

Il concetto dal quale è partito l'autore per sviluppare questo programma è il seguente: utilizzando le istruzioni GET e PRINT del BASIC, il Commodore legge e scrive rispettivamente dal disco-sorgente e sul disco-destinazione le informazioni opportunamente trattate per raggiungere lo scopo finale che è quello di compattare il programma prescelto.
Dopo aver passato tutte le linee per vedere dove si trovano gli eventuali GOTO, GOSUB, ON GOTO, ON GOSUB ed aver posto i relativi numeri di linea in una matrice unidimensionale, riprende a leggere il programma dall'inizio trascrivendolo sul drive destinazione eliminando REM e spazi bianchi e tenendo, naturalmente conto, dei numeri di linea dove aveva trovato i vari salti a routines o subroutines.
Ora esaminiamo riga per riga il listato per una più completa visione del funzionamento.
100-116. Commento iniziale, di nessuna utilità pratica, che contiene informazioni sul programma.
117. Pulizia di tutte le variabili e dimensionamento della matrice.
118-131. Messa a punto dei colori di schermo ed istruzioni per il fun-

zionamento. nella **riga 124**, tra virgolette, è scritta la parola "Compattatore" in caratteri shiftati, ciò per fare in modo che in fase di output su video risultino maiuscoli.
132-140. Assunzione dei vari INPUT richiesti dal programma ed apertura del canale di lettura con controllo di errore del disco (GOSUB 363).
141-149. Dopo aver letto i primi caratteri si controlla che T e T1 siano diversi da zero. In caso affermativo legge il numero di linea che deve trattare per vedere se esistono dei salti incondizionati.
150-159. Lettura dei primi due caratteri della linea e controllo con il codice ASCII 137 o 141, questi numeri corrispondono ai tokens GOTO e GOSUB (i tokens, per chi non lo sapesse, sono i numeri chiavi e delle parole e BASIC). In caso affermativo si esegue la lettura e memorizzazione del numero di linea altrimenti controllo se esiste il token THEN. Se il token viene trovato, stesso procedimento visto per il GOTO, altrimenti lettura di altri due caratteri. Quando la variabile T assumerà il valore 0 il programma ritornerà a leggere un altro numero di linea e quindi il contenuto.
160-170. Se il carattere è uno spazio: lettura di altri due due caratteri; confronto con i codici ASCII compresi tra 48 e 57 per controllare se il valore letto è un numero ed in tal caso memorizzazione del numero nella variabile LT. Quindi lettura di altri due caratteri e ritorno al test per vedere se il carattere è un numero.
171-184. Nel caso in cui il carattere letto non fosse un numero controlla che l'ultimo numero letto non corrisponsa a quello memorizzato prima (naturalmente questo avverrà solamente dopo aver letto almeno un numero); in tal caso salterà ad una routine che controlla i tokens dei salti condizionati. Altrimenti il numero letto viene memorizzato nella matrice TL(N) per poter memorizzare tutti i numeri letti, poiché la variabile LT viene aggiornata ad ogni lettura. Dopo stamperà sul video il numero trattato e andrà a controllare la routine dei salti condizionati.
185-192. Nel caso il carattere letto corrisponda il codice ASCII dei tokens ON GOTO o ON GUSUB ritorno alla routine di lettura e memorizzazione del numero. Altrimenti, lettura dei due caratteri seguenti sino a che non trova uno spazio; in tal caso ritorno al controllo della variabile T = 0 ed a seconda del risultato inizierà il compattamento (T = 0) o riprenderà da capo tutti i controlli sulla linea seguente (T < >0).
193-204. Controllo che il numero delle linee sia maggiore di due ed in tal caso riordinamento secondo il numero di linea.
205-218. Chiusura e riapertura del canale di lettura per iniziare il compattamento; trasformazione del nome del programma originale con l'aggiunta del suffisso '/C' ed apertura del canale di scrittura con controllo di errore del disco. Nel caso in cui esistesse un programma con il medesimo nome, viene eseguita la cancellazione automatica.
219-226. Lettura dei primi due caratteri che contengono il puntatore alla prossima linea e scrittura sul disco destinazione.
228-249. Controllo di fine programma tramite le variabili T e T1 che devono essere uguali a 0; se il programma non è finito stampa sul video del numero di linea del programma che verrà scritto sul disco destinazione e ritorno alla lettura di altri due caratteri. Se i caratteri letti sono uno spazio od un due punti continua la lettura dei caratteri seguenti altrimenti se T è uguale a 0 test sulla matrice TL (N) per i vari GOTO. Prima di prendere altri due caratteri la **linea 240** controlla che il token che segue non sua una REM altrimenti passa alla scrittura del numero di linea sul disco destinazione.
Se il test sulla matrice TL(N) relativo al numero di linea che si sta trattando con il numero di linea del programma sorgente risulta essere uguale allora scrive il numero di linea sul disco destinazione altrimenti ritorna all'inizio della routine per riprendere tutto il ciclo da capo. Nella **linea 249** c'è il controllo della variabile F. Una volta scritto il numero di linea la variabile F viene azzerata e quindi si passa al controllo della linea attuale.
250-261. Controlla di nuovo se i caratteri letti corrispondono a GOTO o IF. ...THEN ed in tal caso pone F uguale ad 1; poi a seconda che T sia uguale a 0 o 32 (spazio) salta al controllo di fine linea o riprende altri due caratteri.
262-270. Se il valore di T era diverso dai due numeri precedenti ed è diverso dal codice ASCII del token REM salta alla **riga 277** altrimenti continua a leggere caratteri (che sono da eliminare perché scritti dopo il token REM) fino a quando T è uguale a 0; in tal caso salta al controllo e scrittura ultimi caratteri della linea.
271-284. Questa routine serve per controllare quando la lettura dei caratteri incontra il codice ASCII 34 (virgolette) per poter permettere di scrivere sul disco destinazione i caratteri racchiusi tra virgolette senza modificarli. Alla fine salta al controllo e scrittura ultimi caratteri della riga.
285-297. Controllo del carattere 'due punti', in tal caso ritorno alla routine della scansione linea BASIC altrimenti legge due caratteri e se T vale 32 (spazio) o 58 (due punti) ritorna a leggerne altri due invece se T è uguale a 143 (REM) ritorna alla routine descritta prima dove elimina tutti i caratteri dopo tale istruzione. Se T è uguale a 0 va alla routine di fine riga altrimenti stampa un due punti per separare una istruzione dall'altra e poi ritorna alla **riga 258** dove riprende il controllo della linea BASIC.
298-308. La **linea 304** controlla che la linea BASIC scritta sul disco destinazione sia maggiore di 170 caratteri oppure che F sia uguale ad 1; in tal caso pone T uguale a 0 e poi ritorna alla lettura di un nuovo numero di linea. Per vedere se il numero dei caratteri è maggiore di 170 durante il programma viene utilizzata la variabile R che viene opportunamente incrementata durante l'esecuzione. Le **linee 305-308** provvedono a leggere due caratteri per vedere se il compattamento

è finito o leggere un nuovo numero di linea.
309-317. Test per sapere se la linea che si è esaminata è finita ed in tal caso salto alla routine di scrittura degli ultimi caratteri della linea.
318-326. Se la linea che si sta esaminando non è finita lettura di altri due caratteri ed a seconda del valore assunto da T legge il prossimo numero di linea (T = 0) o legge altri due caratteri (T = 32 o 58) o salta alla routine per eliminare i caratteri scritti dopo il token REM (T = 143); se T non assume nessuno di questi valori scrive un due punti per separare le istruzioni e ritorna al controllo della linea BASIC.
327-337. Se la condizione testata nella **linea 295** è vera il programma ora scrive gli ultimi caratteri della riga, pone F uguale a 0 e ritorna alla scansione della riga BASIC del programma sorgente.
338-353. Quando nel corso del programma si verifica la condizione che T e T1 sono uguali a 0 significa che il programma sorgente è finito poiché ogni programma è sempre chiuso con due caratteri che corrispondono al codice ASCII 0. Quindi vengono scritti tre zeri (uno per la fine riga e due per fine programma), chiusi i canali di lettura, scrittura e comando (per controllare le routine di errore del disco) e viene chiesto se si vuole compattare un altro programma oppure passare ad altri lavori. In quest'ultimo caso viene eseguito un system reset del Commodore (quello citato nel programma corrispondente al C64; per il VIC 20 eseguire SYS64802, per il Commodore della serie 4000 e 8000 SYS64790).
354-362. Lettura di due caratteri dal disco sorgente e trasformazione in codice ASCII; il valore viene posto nella variabile T e T1.
363-366. Lettura delle variabili che contengono un eventuale errore del disco ed in tal caso interruzione dell'esecuzione del programma e stampa su video dell'errore che si è verificato.

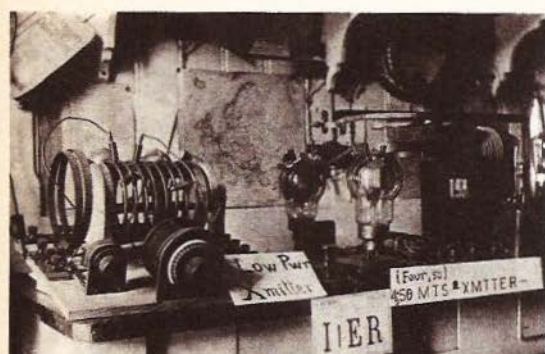
Conclusione

Dato che durante il programma vi sono una infinità di GOTO e GOSUB, alla prima lettura può anche non risultare del tutto chiaro il funzionamento, ma afferrata la chiave logica risulterà tutto molto più chiaro. Penso che il primo programma che proverete a compattare sarà senz'altro questo stesso, data la numerosa presenza di REM che del resto servono per rendere più capibile il listato.
Nel cosro dell'articolo compare un piccolo programma di prova che serve per verificare che il compattatore 'giri', poiché compattare subito un lungo programma senza essere sicuri del suo perfetto funzionamento comporterebbe una notevole perdita di tempo. Nel prossimo numero della rivista comparirà lo scompattatore che fa esattamente il contrario del programma ora proposto. Quindi arrivederci a tutti sul prossimo numero.

```
PROVA

10 PRINT"[CLEAR]"
11 X=0
12 SL=INT(2000*RND(1))
13 IF SL>2023 THEN 12
14 IF SL<1024 THEN 12
15 POKE SL,42
16 POKE SL+54272,INT(15*RND(1))
17 X=X+1
18 IF X>400 THEN 20
19 GOTO 12
20 FOR D=1 TO 800
21 NEXTD
22 PRINT"[CLEAR]"
23 PRINT"PREMERE [RVS]F[RVOFF] PE
   R FINIRE"
24 PRINT"PREMERE [RVS]R[RVOFF] PE
   R RICOMINCIARE"
25 GET AB$
26 IF AB$="" THEN 25
27 IF AB$="F" THEN 30
28 IF AB$="R" THEN 10
29 GOTO 25
30 END
```

```
PROVA/C

10 PRINT"[CLEAR]":X=0
12 SL=INT(2000*RND(1)):IF SL>2023
   THEN 12
14 IF SL<1024 THEN 12
15 POKE SL,42:POKE SL+54272,INT(1
   5*RND(1)):X=X+1:IF X>400 THEN
   20
19 GOTO 12
20 FOR D=1 TO 800:NEXTD:PRINT"[CL
   EAR]":PRINT"PREMERE [RVS]F[RVO
   FF] PER FINIRE":PRINT"PREMERE
   [RVS]R[RVOFF] PER RICOMINCIARE
   "
25 GET AB$:IF AB$="" THEN 25
27 IF AB$="F" THEN 30
28 IF AB$="R" THEN 10
29 GOTO 25
30 END
```

```
100 REM *************************
101 REM *                       *
102 REM *C O M P A T T A T O R E*
103 REM *                       *
104 REM *      MODIFICA DI      *
105 REM *                       *
106 REM *       GIANCARLO       *
107 REM *                       *
108 REM *      DE  COBELLI      *
109 REM *                       *
110 REM *   V.LE DEI FIORI 65   *
111 REM *                       *
112 REM *  CUSANO MILANINO  MI  *
113 REM *                       *
114 REM *    TEL. 02/6192306    *
115 REM *                       *
116 REM *************************
117 CLR :DIM TL(1000)
118 REM *************************
119 REM *                       *
120 REM *      ISTRUZIONI       *
121 REM *                       *
122 REM *************************
123 POKE 53280,0:POKE 53281,0:PRIN
    TCHR$(14)
124 PRINT"[VERDE][CLEAR]" TAB(12)"
    [RVS] ┬╲♠||♠|┌_¯ [2 DOWN]
125 PRINT"●UESTA UTILITY COMPATTA
    UN PROGRAMMA PO-
126 PRINT"STO NEL DRIVE PRESCELTO
    CANCELLANDO TUT-
127 PRINT"TE LE  REM, ANNOTAZIONI,
      SPAZI  VUOTI E
128 PRINT"LINCANDO TUTTE LE LINEE
    POSSIBILI.[2 DOWN]
129 PRINT"¬L PROGRAMMA COMPATTATTO
      AVRA' LO STESSO
130 PRINT"NOME, MA SEGUITO DAL SUF
    FISSO '/C' E SA-
131 PRINT"RA' SALVATO SUL DRIVE PR
    ESCELTO.[DOWN]"
132 INPUT "[RVS] ¯RIVE SORGENTE
     #";DO$
133 IF DO$<CHR$(48) OR DO$>CHR$(49
    ) THEN PRINT"[UP]":GOTO 132
134 INPUT "[RVS] ¯RIVE DESTINAZION
    E#";DD$
135 IF DD$<CHR$(48) OR DD$>CHR$(49
    ) THEN 134
136 INPUT "[RVS] ╱OME PROGRAMMA
      ";NP$
137 OPEN 15,8,15
138 OPEN 5,8,5,DO$+":"+NP$+",P,R":
    GOSUB 363
139 PRINT"[RVOFF][CLEAR]●CANSIONE
    PROGRAMMA
140 PRINT"   PER LINEE DA COMPATTA
    RE....[2 DOWN]"
141 REM *************************
142 REM *                       *
143 REM * LETTURA  PUNTATORE E  *
144 REM *  NUMERO DELLA LINEA   *
145 REM *                       *
146 REM *************************
147 GOSUB 363:GOSUB 359
148 GOSUB 359:IF T+T1=0 THEN 199
149 GOSUB 359:LN=T1+(256*T)
150 REM *************************
151 REM *                       *
152 REM * SCANSIONE LINEE BASIC *
153 REM *   PER  GOTO   GOSUB   *
154 REM *                       *
155 REM *************************
156 GOSUB 360
157 IF T=0 THEN 148
158 IF T=137 OR T=141 THEN 166
159 IF T<>167 THEN 156
160 REM *************************
161 REM *                       *
162 REM *  MEMORIZZAZIONE LINEE *
163 REM *     DA  COMPATTATARE  *
164 REM *                       *
165 REM *************************
166 LT=0
167 GOSUB 360:IF T=32 THEN 167
168 IF T<48 OR T>57 THEN 177
169 LT=(10*LT)+VAL(C$)
170 GOSUB 360:GOTO 168
171 REM *************************
172 REM *                       *
```

```
173 REM *  TEST PER CONTROLLARE *
174 REM *    SE GIA'  TROVATA    *
175 REM *                        *
176 REM *************************
177 FOR X=0 TO N
178 IF TL(X)=LT THEN 190
179 NEXTX
180 TL(N)=LT:N=N+1
181 PRINTLT,
182 IF N<1000 THEN 190
183 PRINT"[2 DOWN]TROPPE LINEE DA
    COMPATTARE!"
184 GOTO 345
185 REM *************************
186 REM *                        *
187 REM * TEST PER ONGOTO/GOSUB *
188 REM *                        *
189 REM *************************
190 IF T=44 THEN 166
191 IF T<>32 THEN 157
192 GOSUB 360:GOTO 190
193 REM *************************
194 REM *                        *
195 REM *   RIORDINO LINEE DA    *
196 REM *       COMPATTARE       *
197 REM *                        *
198 REM *************************
199 IF N<2 THEN 211
200 FOR X=0 TO N-1
201 FOR Y=0 TO N-2
202 IF TL(Y)<TL(X) THEN 204
203 T=TL(Y):TL(Y)=TL(X):TL(X)=T
204 NEXTY,X
205 REM *************************
206 REM *                        *
207 REM *INIZIALIZZAZIONE PER IL*
208 REM *     COMPATTAMENTO      *
209 REM *                        *
210 REM *************************
211 PRINT"[CLEAR]♥TO COMPATTANDO..
    ...[2 DOWN]
212 CLOSE 5
213 OPEN 5,8,5,DO$+":"+NP$+",P,R"
214 GOSUB 363
215 ND$=LEFT$(NP$,14)+"/C"
216 PRINT#15,"S"+DD$+":"+ND$
217 OPEN 6,8,6,DD$+":"+ND$+",P,W"
218 GOSUB 363
219 REM *************************
220 REM *                        *
221 REM *   COPIA PUNTATORE DI   *
222 REM *          LINEA         *
```

```
223 REM *                        *
224 REM *************************
225 GOSUB 359
226 PRINT#6,CHR$(T1);
227 PRINT#6,CHR$(T);:R=0
228 REM *************************
229 REM *                        *
230 REM *COPIA LINEA E NUM.LINEA*
231 REM *                        *
232 REM *************************
233 GOSUB 359:K1=T1:K2=T
234 F=0:IF T+T1=0 THEN 344
235 GOSUB 359:L1=T1:L2=T
236 LN=L1+(256*L2):PRINT LN,
237 GOSUB 360
238 IF T=32 OR T=58 THEN 237
239 IF T=0 THEN 242
240 IF T<>143 THEN 246
241 GOSUB 360:IF T>0 THEN 241
242 F=1:FOR X=0 TO N
243 IF TL(X)<LN THEN NEXTX
244 IF TL(X)=LN THEN 246
245 GOTO 233
246 PRINT#6,CHR$(K1);CHR$(K2);
247 PRINT#6,CHR$(L1);CHR$(L2);:R=4
248 IF F THEN PRINT#6,":";:R=5
249 F=0:GOTO 258
250 REM *************************
251 REM *                        *
252 REM *SCANSIONE LINEE BASIC E*
253 REM *COMPATTAMENTO PROGRAMMA*
254 REM *                        *
255 REM *************************
256 PRINT#6,C$;:R=R+1
257 GOSUB 360
258 IF T=137 THEN F=1
259 IF T=139 OR T=167 THEN F=1
260 IF T=0 THEN 304
261 IF T=32 THEN 257
262 REM *************************
263 REM *                        *
264 REM *ELIMINAZIONE 'REM'  DAL*
265 REM *   RESTO DELLA LINEA    *
266 REM *                        *
267 REM *************************
268 IF T<>143 THEN 277
269 GOSUB 360:IF T>0 THEN 269
270 GOTO 304
271 REM *************************
272 REM *                        *
273 REM *COPIA FINO AL  PROSSIMO*
274 REM *    REM  O FINE LINEA   *
```

```
275 REM *                        *
276 REM **************************
277 IF T<>34 THEN 291
278 PRINT#6,C$;:R=R+1
279 GOSUB 360
280 IF T=34 THEN 256
281 IF T>0 THEN 278
282 IF F THEN T=0:GOTO 227
283 PRINT#6,CHR$(34);:R=R+1
284 GOTO 304
285 REM **************************
286 REM *                        *
287 REM *TEST ':' FINO  AL PROS-*
288 REM *SIMO CARATTERE  E COPIA*
289 REM *                        *
290 REM **************************
291 IF T<>58 THEN 256
292 GOSUB 360
293 IF T=32 OR T=58 THEN 292
294 IF T=143 THEN 269
295 IF T=0 THEN 304
296 PRINT#6,":";:R=R+1
297 GOTO 258
298 REM **************************
299 REM *                        *
300 REM *    TEST FINE LINEA     *
301 REM *    E COMPATTAMENTO     *
302 REM *                        *
303 REM **************************
304 IF F  OR (R>170) THEN T=0:GOTO
    227
305 GOSUB 359
306 IF T+T1=0 THEN 344
307 GOSUB 359:LN=T1+(256*T)
308 L1=T1:L2=T:PRINT LN,
309 REM **************************
310 REM *                        *
311 REM * TEST  CONTROLLO LINEA *
312 REM *   SE GIA'COMPATTATA    *
313 REM *                        *
314 REM **************************
315 FOR X=0 TO N
316 IF TL(X)<LN THEN NEXTX
317 IF TL(X)=LN THEN 332
318 REM **************************
319 REM *                        *
320 REM *SCARTO LINEE  NON USATE*
321 REM *                        *
322 REM **************************
323 GOSUB 360:IF T=143 THEN 269
324 IF T=32 OR T=58 THEN 323
325 IF T=0 THEN 305
326 PRINT#6,":";:R=R+1:GOTO  258
327 REM **************************
328 REM *                        *
329 REM * SCRITTURA ULTIMA RIGA *
330 REM *      IN  QUESTIONE     *
331 REM *                        *
332 REM **************************
333 PRINT#6,CHR$(0);CHR$(1);CHR$(1
    );
334 PRINT#6,CHR$(L1);CHR$(L2);:R=4
335 GOSUB 360:IF T=32 OR T=58 THEN
     334
336 IF T=0 OR T=143 THEN PRINT#6,"
    :";
337 F=0:GOTO 258
338 REM **************************
339 REM *                        *
340 REM *FINE DEL  COMPATTAMENTO*
341 REM *CHIUSURA DEL  PROGRAMMA*
342 REM *                        *
343 REM **************************
344 PRINT#6,CHR$(0);CHR$(0);CHR$(0
    );
345 CLOSE 5:CLOSE 6:CLOSE 15
346 PRINT"[CLEAR]":PRINT TAB(10):P
    RINT"[4 DOWN]-OMPATTAMENTO  FI
    NITO"
347 FOR K=1 TO 500:NEXTK
348 PRINT"[CLEAR]\L PROGRAMMA COMP
    ATTATO E' STATO SALVATO"
349 PRINT"SUL DISCO NEL DRIVE PRES
    CELTO."
350 INPUT "[5 DOWN]XUOI COMPATTARE
     UN ALTRO PROGRAMMA([RVS]S/N[R
    VOFF])";SN$
351 PRINT"[CLEAR]"
352 IF SN$="S" THEN 132
353 SYS64738
354 REM **************************
355 REM *                        *
356 REM *      SUBROUTINES       *
357 REM *                        *
358 REM **************************
359 GOSUB 360:T1=T
360 GET #5,C$:GOSUB 363
361 IF C$="" THEN T=0:RETURN
362 T=ASC(C$):RETURN
363 INPUT#15,EN,EM$,ET,ES
364 IF EN=0 THEN RETURN
365 PRINT"[4 DOWN][RVS]-RRORE NEL
    DISCO[DOWN]"
366 PRINTEN;EM$;ET;ES
```

SPOSTA BASIC E ALTRO

di **Ernesto Sidoti**

PROBLEMA 1

Il chip 6567, detto anche VIC II, può "leggere" per ogni sprite un gruppo di DATA allocati in locazioni prefissate dall'utente. Quando si vuole accendere sullo schermo una sprite, bisogna far sapere al VIC II dove andare a reperire le informazioni che creeranno l'immagine. Le locazioni dove si memorizzano gli indirizzi vanno da 2040 a 2047.
In questi registri, chiamati puntatori della sprite, si dovrà memorizzare un sottomiltiplo dell'indirizzo della prima cella di memoria dove risiedono le informazioni per definire una sprite.
Ad esempio se nella locazione 2040, che è il puntatore della sprite 0, mettiamo il valore 13, il VIC II andrà a leggere i valori a partire dalla locazione 832 per finire alla locazione 895. Tale indirizzo si ricava dal prodotto del valore posto nel puntatore, (13 in questo caso) e col numero di bytes necessari per la creazione di una sprite 63 + 1. L'ultimo valore non viene considerato, dato che il Commodore 64 usa per i propri calcoli il sistema binario. (64 è uguale a 2 elevato a 6).

INDIRIZZO BLOCCO 13*64 = 832

Fatta questa piccola premessa, ti dovresti già essere accorto che il VIC II può accedere fino alla locazione 16320 (255*64). (A = B*C, dove B è il massimo valore che il puntatore può assumere e C è il numero di valori necessari per definire una sprite.)
All'interno dell'area RAM, accessibile dai puntatori delle sprite, molte locazioni sono inutilizzabili per archiviare i DATA.

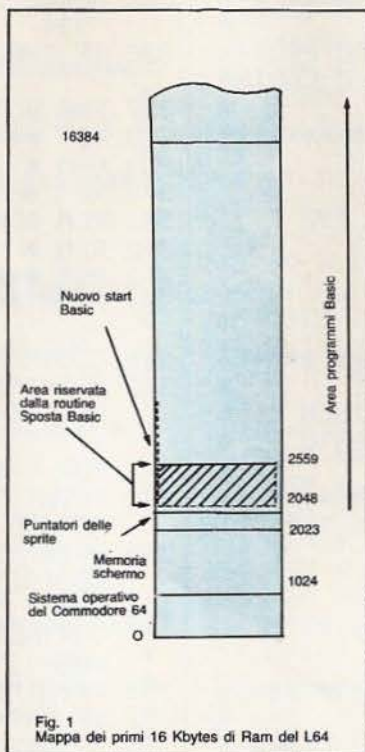



Fig. 1
Mappa dei primi 16 Kbytes di Ram del L64

Numero sprite	Registro X	Registro Y
0	53248	53249
1	53250	53251
2	53252	53253
3	53254	53255
4	53256	53257
5	53258	53259
6	53260	53261
7	53262	53263

Fig. 2
Registri di posizione per ogni sprite

È oramai noto che il primo K byte di RAM è usato dal sistema operativo del COMMODORE 64: è quella parte cioè che viene definita pagina zero. Quindi il solo spazio disponibile è quello che va dalla locazione 828 fino a 1023. Questo spazio è assegnato al buffer di cassetta, quindi inutilizzabile quando si usa il registratore. La sola altra area disponibile è quella utilizzata dal testo BASIC il cui inizio, come si sa, è fissato a 2048 e si estende fino a 40959, rendendo disponibile ai tuoi programmi BASIC 38911 Bytes.
Il problema a tal punto dovrebbe essere chiaro. Dove archiviare una sequenza di DATA senza paura che venga modificata dal sistema operativo o dal testo di un programma in BASIC?
In questo articolo propongono una routine capace di portare più in alto l'indirizzo di partenza del programma BASIC che da 2048 andrà a fissarsi a 2560, lasciando così 512 Bytes liberi per l'allocazione dei DATA per le sprite. (vedi fig. 1)
Dopo aver digitato la routine basta dare il seguente comando:
SYS49329
e lo start del BASIC sarà allocato a 2560. Il BASIC manterrà tutte le caratteristiche che aveva prima che avvenisse lo spostamento in alto, cioè caricherà normalmente dalle periferiche e accetterà senza problemi caratteri da tastiera.

PROBLEMA 2

Per poter posizionare le sprite sullo schermo vengono usati 16 registri allocati da 53248 a 53263. In queste locazioni sono memorizzati i dati riguardanti la posizione X e Y

di ogni sprite. (vedi fig. 2).
Ricordati che i registri (X e Y) memorizzano la posizione del pixel superiore sinistro. Sullo schermo del tuo C64 si possono accendere 320 pixel sull'asse X e 200 su quello Y, ma per le sprite l'area visibile sull'asse orizzontale va da 24 fino a 344 e da 50 sino a 250 per quello verticale. (vedi fig. 4).

Esistono per la sprite posizioni anche invisibili: tali luoghi sono necessari per permettere alla sprite di uscire dall'area visibile completamente senza apparire dal margine opposto. Come ben saprai, in ogni locazione il valore massimo che puoi memorizzare è 255, ma la posizione X della sprite può andare ben oltre a tale valore. L'inconveniente si risolve usando il registro MSBX allocato in 53265.
Così facendo potremo disporre per ogni sprite oltre agli 8 bit del registro X un bit del registro MSBX realizzando così un Byte di nove bit.

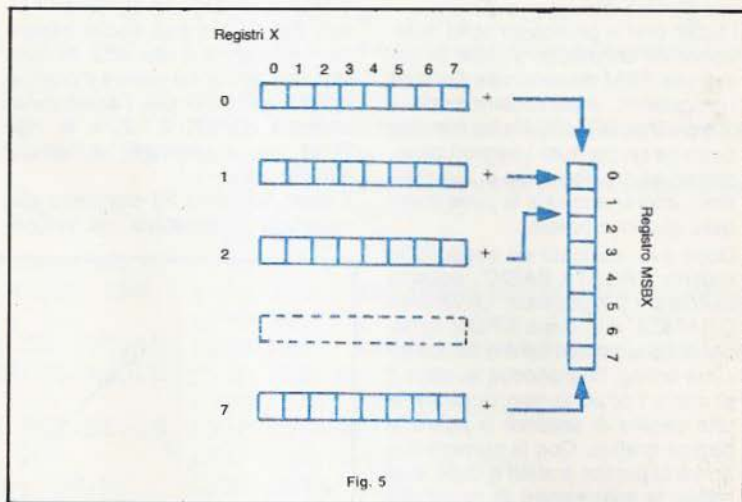



Fig. 5

Con l'espediente possiamo superare agevolmente il valore di 255.
Un simile sistema di posizionamento può risultare un po' complicato perché per valori di X compresi da 0 a 255 si usa solo il registro X, mentre per valori superiori anche un bit del registro MSBX, (fig. 5). Se ad esempio vogliamo posizionare la sprite sulla posizione 256 basterà portare a 0 il valore del corrispondente registro X e a 1 il valore del registro MSBX.
Per agevolare l'uso di questi registri basta usare la routine SPOSTA BASIC: questa associa a X% e Y% la posizione della sprite sullo schermo e a SN% il numero della sprite da considerare. Con questa routine non sarà più necessario usare i registri di X, Y e MSBX in quanto le posizioni saranno direttamente assegnate alle variabili sopra citate. Per richiamare la routine, sempre dopo averla caricata in memoria, basta assegnare i valori X% e Y% e SN% da 0 7) e poi dare:
SYS49217
Per maggiore chiarezza, osserva e prova i programmi esplicativi 'PROVA SPRITE' e 'LAVAGNA GRAFICA'.
Nel primo programma 'PROVA SPRITE' nelle righe che vanno dalla 320 alla 500 c'è un chiaro esempio di come portare a spasso due sprite.
'LAVAGNA GRAFICA' è un programma che oltre a essere esplicativo per l'uso della routine qui presentata, ti permette di disegnare in alta risoluzione e archiviare su di-

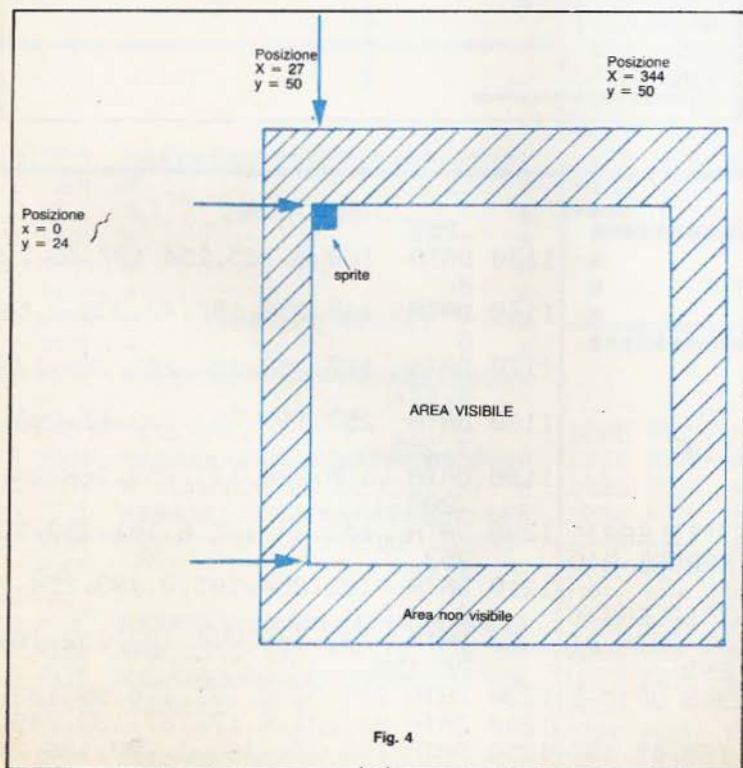
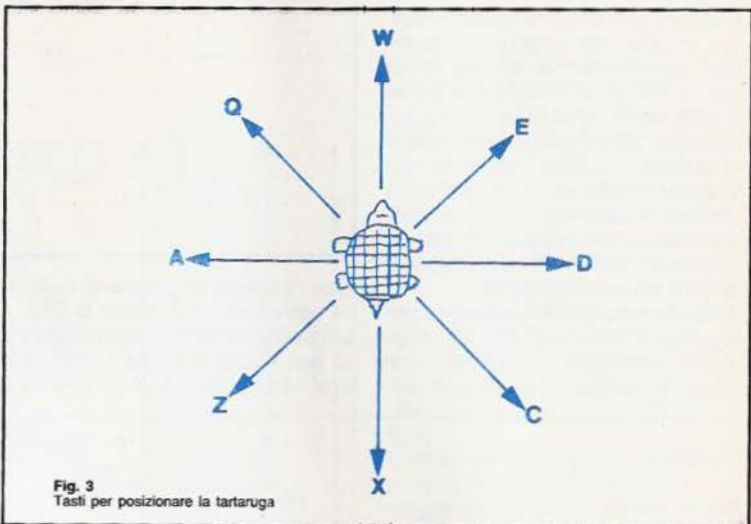



Fig. 4

sco l'intero disegno realizzato.
I listati che vi propongo sono notevolmente chiarificati tramite le numerose REM disseminate durante i programmi, in particolare modo la 'LAVAGNA GRAFICA' che racchiude in se un po' tutti i segreti esposti fino ad ora. Per l'uso di quest'ultimo programma vale la pena spendere qualche parola.

Dopo aver caricato ed eseguito la routine 'SPOSTA BASIC', occorre caricare il programma 'LAVAGNA GRAFICA' e.... dopo il RUN apparirà sullo schermo un menù con alcune scelte. Digitando la numero 1 si attiva l'opzione con la quale in una decina di secondi si pulirà la pagina grafica. Con la numero 2 si aprirà la pagina grafica e dopo aver atteso la colorazione di quest'ultima premere un tasto di funzione che farà apparire una piccola tartaruga al centro dello schermo. Muovendo opportunamente questa tartaruga su e giù con i tasti definitivi e riportati in figura 3 ti sarà permesso di realizzare disegni in alta risoluzione sullo schermo del tuo COMMODORE 64. Con le scelte 3 e 4 si può scrivere o leggere da periferica il disegno archiviato. Nel programma, utilizzo una periferica a dischi floppy. Ma può anche essere un registratore a cassette. In questo caso dovranno essere modificate: la riga 1910 per l'apertura in scrittura (OPEN 2,1,2) e la riga 2040 per l'apertura in lettura (OPEN 2,1).
Il tasto funzione F1 permette alla tartaruga di spostarsi più velocemente o più lentamente: permettendolo una volta si otterrà l'accellerazione e premendolo una seconda volta si rallenterà'. Il tasto funzione F2 permette di fare delle correzioni sulle linee già tracciate. Riprendendolo si trova a disegnare. F3 Premendo questo tasto la tartaruga si sposterà senza tracciare o cancellare.

Fig. 3
Tasti per posizionare la tartaruga

```
1000 REM  **************************
1010 REM  *                        *
1020 REM  *      SPOSTA BASIC      *
1030 REM  *                        *
1040 REM  **E********************S***
1050 :
1060 PRINT"[CLEAR]"
1070 L=49152 :H=0
1080 READ T:IF T=-2 THEN 1110
1090 POKE L,T:L=L+1
1100 H=H+T:GOTO 1080
1110 IF H<>26366 THEN PRINT"[CLEAR][
     5 DOWN]CONDIZIONE DI ERRORE RIC
     ONTROLLA I DATA":END
1120 IF L<>49350 THEN PRINT"[CLEAR][
     RIGHT][4 DOWN]NUMERO DI DATA DI
     VERSO DAL RICHIESTO":END
1130 PRINT"[2 DOWN]SYS49329[3 UP]":E
     ND
1140 DATA  165,46,133,254,165,45,133
     ,253
1150 DATA  160,0,165,254,197,48,208,
     8
1160 DATA  165,253,197,47,208,2,56,9
     6
1170 DATA  177,253,197,252,208,17,20
     0,177
1180 DATA  253,197,251,208,11,200,17
     7,253
1190 DATA  170,200,177,253,168,24,96
     ,200
1200 DATA  152,24,105,6,101,253,133,
     253
1210 DATA  165,254,105,0,133,254,76,
     8
1220 DATA  192,169,211,133,252,169,2
     06,133
1230 DATA 251,32,0,192,176,89,152,48
1240 DATA 91,201,8,176,87,133,249,10
1250 DATA 133,250,169,1,200,136,240,
```

```
    4
1260 DATA 10,76,93,192,133,248,169,2
    17
1270 DATA 133,252,169,128,133,251,32
    ,0
1280 DATA 192,176,52,166,250,152,157
    ,1
1290 DATA 208,169,216,133,252,169,12
    8,133
1300 DATA 251,32,0,192,176,33,138,72
1310 DATA 166,250,152,157,0,208,104,
    201
1320 DATA 0,208,11,165,248,73,255,45
1330 DATA 16,208,141,16,208,96,165,2
    48
1340 DATA 13,16,208,141,16,208,96,16
    2
1350 DATA 11,108,0,3,162,14,108,0
1360 DATA 3,162,10,160,0,24,32,156
1370 DATA 255,200,132,43,134,44,169,
    0
1380 DATA 141,0,10,76,66,166,-2
```

```
1000 REM  ************************
1010 REM  *                      *
1020 REM  *    PROVA SPRITE      *
1021 REM  *                      *
1022 REM  *         DI           *
1023 REM  *                      *
1024 REM  *   ERNESTO SIDOTI     *
1030 REM  *                      *
1040 REM  **E*****************S**
1050 :
1060 PRINT"[CLEAR]"
1070 REM  ************************
1080 REM  * CARICO IL BLOCCO 32  *
1090 REM  ************************
1100 FOR A=2048 TO 2111:POKE A,255:N
    EXT
1110 :
1120 REM  ************************
1130 REM  *ABILITO LE SPRITE 0 E 1*
1140 REM  ************************
1150 POKE 53269,3
1160 :
1170 REM  ************************
1180 REM  *  STABILISCO I ¬I  *
1190 REM  ************************
1200 POKE 53287,1
1210 POKE 53288,0
1220 REM  ************************
1230 REM  *  CARICO I  PUNTATORI *
1240 REM  ************************
1250 POKE 2040,32
1260 POKE 2041,32
1270 :
1280 REM  ************************
1290 REM  *    MUOVO LE SPRITE   *
1300 REM  ************************
1310 FOR A=25 TO 319 STEP 2
1320 X%=A:Y%=100:SN%=0
1330 SYS49217
1340 X%=A:Y%=150:SN%=1
1350 SYS49217
1360 NEXT
1370 FOR A=319 TO 25 STEP -2
1380 X%=A:Y%=150:SN%=1
1390 SYS49217
1400 NEXT
1410 FOR A=319 TO 24 STEP -2
1420 X%=A:Y%=100:SN%=0
1430 SYS49217
1440 NEXT
1450 GOTO 1300
1460 X%=A:Y%=100:SN%=1
1470 SYS49217
1480 NEXT
1490 GOTO 1300
```

```
1000 PRINT"[CLEAR]"
1010 REM  *************************
1020 REM  *                       *
1030 REM  *    LAVAGNA GRAFICA    *
1040 REM  *                       *
1050 REM  *          DI           *
1051 REM  *                       *
1052 REM  *   DI ERNESTO SIDOTI   *
1060 REM  *                       *
1070 REM  ***E*****************S***
1080 :
1090 REM  *************************
1091 REM  *                       *
```

```
1100 REM  *  CARICO NEL BLOCCO 32   *
1101 REM  *                         *
1110 REM  **************************
1120 FOR A=2048 TO 2110:READ B:POKE
     A,B:NEXT
1130 :
1140 REM  **************************
1141 REM  *                         *
1150 REM  *   ABILITO LA SPRITE 0   *
1151 REM  *                         *
1160 REM  **************************
1170 POKE 53269,1
1180 :
1190 REM  **************************
1191 REM  *                         *
1200 REM  * DEFINISCO IL PUNTATORE *
1201 REM  *                         *
1210 REM  **************************
1220 POKE 2040,32
1230 :
1240 LI=8192
1250 X=160:Y=100:S=1:D=1
1260 REM  **************************
1261 REM  *                         *
1270 REM  *   RIPETIZIONE TASTI     *
1271 REM  *                         *
1280 REM  **************************
1290 POKE 650,128
1300 :
1310 GOTO 2180
1320 REM  **************************
1321 REM  *                         *
1330 REM  * PULISCE PAGINA GRAFICA *
1331 REM  *                         *
1340 REM  **************************
1350 FOR K=LI TO LI+7999:POKE K,0:NE
     XT
1360 PRINT"[CLEAR][10 DOWN][6 RIGHT]
     FINITO !!!":FOR A=1 TO 1000:NEX
     T:GOTO 2180
1370 REM  **************************
1371 REM  *                         *
1380 REM  *  APRE PAGINA GRAFICA    *
1381 REM  *                         *
1390 REM  **************************
1400 POKE 53269,1:POKE 53272,PEEK(53
     272) OR 8
1410 POKE 53265,PEEK(53265) OR 32
1420 FOR A=1024 TO 2023:POKE A,16:NE
     XT
1430 GOSUB 1660
1440 REM  **************************
1441 REM  *                         *
1450 REM  *     MUOVO LA SPRITE     *
1451 REM  *                         *
1460 REM  **************************
1470 Z=X+12:Q=Y+40
1480 X%=Z:Y%=Q:SN%=0
1490 SYS49217
1500 IF H=1 THEN 1430
1510 REM  **************************
1511 REM  *                         *
1520 REM  *    TRACCIO I PUNTI      *
1521 REM  *                         *
1530 REM  **************************
1540 P=LI+INT(Y/8)*320+8*INT(X/8)+(Y
     AND 7)
1550 POKE P,(PEEK(P) OR (2↑(7-(X AND
     7))))*(-1*F)
1560 GOTO 1430
1570 REM  **************************
1571 REM  *                         *
1580 REM  *   DATA PER LA SPRITE    *
1581 REM  *                         *
1590 REM  **************************
1600 DATA 0,56,0,0,124,0,0,84,0,0
1610 DATA 254,0,7,255,192,255,255,25
     5,255,60
1620 DATA 255,126,24,126,30,24,120,3
     0,0,120
1630 DATA 31,195,248,14,0,112,14,24,
     112,254
1640 DATA 60,255,255,255,255,63,255,
     252,3,255
1650 DATA 128,0,254,0,0,56,0,0,56,0,
     0,16,0
1660 GET C$:IF C$="" THEN 1660
1670 IF C$=CHR$(133) THEN JJ=( NOT J
     J):VV=JJ*-1:D=1+(4*VV):S=1+(9*V
     V):J=7
1680 IF C$=CHR$(134) THEN F=( NOT F)
     :J=5
1690 IF C$=CHR$(135) THEN T1=( NOT T
     1):H=T1*-1:J=1
1700 IF C$="←" THEN 1890
1710 IF C$="D" THEN X=X+S
1720 IF C$="A" THEN X=X-S
1730 IF C$="X" THEN Y=Y+S
1740 IF C$="W" THEN Y=Y-S
1750 IF C$="Q" THEN X=X-D:Y=Y-D
1760 IF C$="C" THEN X=X+D:Y=Y+D
1770 IF C$="E" THEN X=X+D:Y=Y-D
1780 IF C$="Z" THEN X=X-D:Y=Y+D
1790 IF Y<0 THEN Y=199
```

```
1800 IF X>320 THEN X=0
1810 IF X<0 THEN X=320
1820 IF Y>199 THEN Y=0
1830 POKE 53287,J
1840 RETURN
1850 REM  *************************
1851 REM  *                       *
1860 REM  * DISABILITO IL VIDEO  E *
1861 REM  *                       *
1870 REM  *  SCRIVO SU PERIFERICA *
1871 REM  *                       *
1880 REM  *************************
1890 POKE 53272,21:POKE 53265,27
1900 POKE 53269,0:PRINT"[CLEAR]":GOT
     O 2180
1910 OPEN 2,8,2,"@:"+M$+",S,W"
1920 POKE 53265,PEEK(53265) AND 239
1930 FOR A=8192 TO 16191
1940 V=PEEK(A)
1950 PRINT#2,VCHR$(13);
1960 NEXT
1970 CLOSE 2
1980 POKE 53265,PEEK(53265) OR 16
1990 RETURN
2000 REM  *************************
2001 REM  *                       *
2010 REM  * DISABILITO IL VIDEO  E *
2011 REM  *                       *
2020 REM  *  LEGGO  DA PERIFERICA *
2021 REM  *                       *
2030 REM  *************************
2040 OPEN 2,8,2,""+M$+",S,R"
2050 POKE 53265,PEEK(53265) AND 239
2060 FOR A=8192 TO 16191
2070 INPUT#2,V
2080 POKE A,V
2090 NEXT
2100 CLOSE 2
2110 POKE 53265,PEEK(53265) OR 16
2120 RETURN
2130 REM  *************************
2140 REM  *                       *
2150 REM  *    CREAZIONE MENU'    *
2160 REM  *                       *
2170 REM  *************************
2180 PRINT"[CLEAR]"
2190 POKE 53280,12 :POKE 53281,12
2200 PRINT"[MARR]    ┌┐          ┌┐"
2210 PRINT"  ┌┴┴──────┴┴┐"
2220 PRINT"  │┌────────┐│"
2230 FOR N=1 TO 8
2240 PRINT"  ││[RVS][NERO]           [
     RVOFF][MARR]││"
2250 NEXT
2260 PRINT"  │└──────────┘│"
2270 PRINT"  ├────────────┤"
2280 PRINT"  ──┬┬───┬┬───┬┬──"
2290 FOR N=1 TO 7
2300 PRINT"     ││        ││  "
2310 NEXT
2320 PRINT"     └┘        └┘  "
2330 PRINT"[4 UP]──┤├──┤├──┤├────
     ───────────────────"
2340 PRINT"[6 UP][7 RIGHT]││"
2350 PRINT"[7 RIGHT]││"
2360 PRINT"[7 RIGHT]││"
2370 PRINT"[7 RIGHT]││"
2380 PRINT"[7 RIGHT]││"
2390 PRINT"[UP][7 RIGHT]││"
2400 PRINT"[7 RIGHT]└┘"
2410 PRINT"[HOME]":U=16
2420 PRINT TAB(U)"1) PULISCE PAGINA"
     :PRINT
2430 PRINT TAB(U)"2) APRE PAGINA":PR
     INT
2440 PRINT TAB(U)"3) LEGGE DA PERIFE
     RICA":PRINT
2450 PRINT TAB(U)"4) SCRIVE SU PERIF
     ERICA":PRINT
2460 PRINT TAB(U)"←) ESCE DA PAG. GR
     AFICA":PRINT
2470 PRINT TAB(U)"F1) AVANZAMENTO VE
     L/LEN":PRINT
2480 PRINT TAB(U)"F3) CANCELLAZIONE"
     :PRINT
2490 PRINT TAB(U)"F5) ALZA/ABBAS PEN
     NELLO":PRINT
2500 PRINT"[HOME]"
2510 PRINT"[4 RIGHT][5 DOWN][RVS][NE
     RO]LAVAGNA"
2520 PRINT"[5 RIGHT][DOWN][RVS][NERO
     ]GRAFICA[RVOFF][MARR]"
2530 GET K$:IF K$="" THEN 2530
2540 IF K$="1" THEN 1350
2550 IF K$="2" THEN 1400
2560 IF K$="3" THEN 2600
2570 IF K$="4" THEN 2590
2580 GOTO 2530
2590 INPUT "[CLEAR][2 RIGHT][7 DOWN]
     SCRIVI IL NOME DEL FILE ";M$:GO
     SUB 1910:GOTO 2180
2600 INPUT "[CLEAR][2 RIGHT][7 DOWN]
     SCRIVI IL NOME DEL FILE ";M$:GO
     SUB 2040:GOTO 2180
```

SUPERLIFE

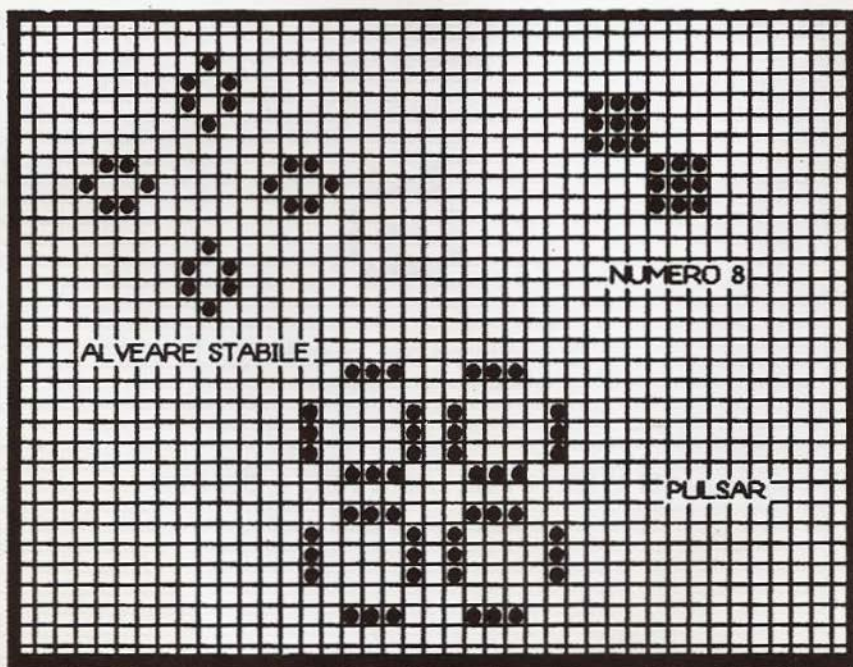
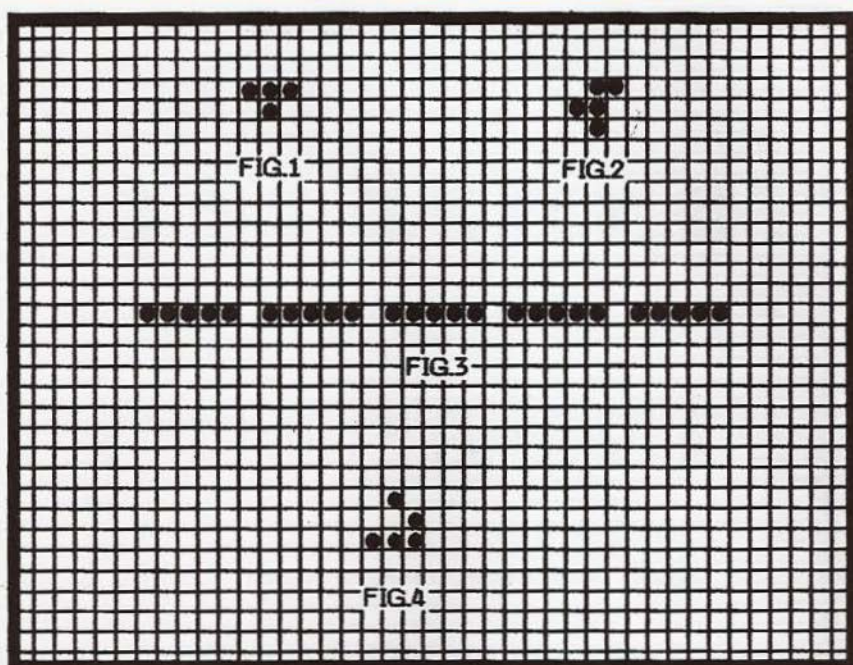
di Marco De Rosa

Life è un gioco di simulazione inventato da John Conway, matematico all'università di Cambrige.
L'idea base è di cominciare con una semplice configurazione di organismi, osservando come questa configurazione iniziale si trasforma applicando le leggi genetiche di Conway.
Le leggi sono le seguenti:
1. **Sopravvivenza**: ogni organismo con due o tre vicini sopravvive fino alla generazione seguente.
2. **Morte**: ogni organismo con quattro o più vicini muore per sovrappopolazione. Ogni organismo con un solo o nessun vicino muore per isolamento.
3. **Nascita**: in ogni cella vuota con esattamente tre vicini nasce un nuovo organismo.
Queste regole sono state scelte da Conway molto accuratamente dopo parecchi esperimenti, in modo da soddisfare i requisiti seguenti:
a) non deve esserci una configurazione iniziale per la quale esista una semplice dimostrazione della possibilità di un aumento illimitato della popolazione.
b) Devono esistere configurazioni iniziali che evidentemente aumentino oltre ogni limite.
c) Devono esistere semplici configurazioni iniziali che crescono e si modificano per un periodo di tempo considerevole prima di giungere a una fine di uno dei tre modi seguenti: estinzione completa (per sovrappopolazione o per troppa rarefazione), stabilizzazione in una configurazione immutabile, subentro di una fase oscillatoria che ripete un ciclo infinito di due o più periodi.
Vediamo ora cosa succede esaminando configurazioni semplici un organismo isolato o una qualsiasi coppia si estinguono alla prima mossa. Le cinque configurazioni di tre cellule si estinguono o giungono ad una configurazione stabile in una generazione. Le configurazioni di quattro elementi sono molto più interessanti. Provate ad esempio il tetromino di figura 1, che giunge ad uno stato stabile in 9 generazioni.
Le previsioni si fanno più complicate quando si arriva a configurazioni di 5 elementi. È da provare il pentomino "erre" di figura 2, che impiega ben 1103 generazioni prima di stabilizzarsi (ma uscirà dallo schermo del vostro CBM 64 prima che riusciate ad arrivarci). Configurazioni ancora più belle da vedere sullo schermo sono le catene di n elementi (fig. 3). Provate per esempio la 5-5-5-5-5. Un'altra splendida scoperta di Conway sono gli alianti, e cioè delle figure che si spostano sullo schermo ricostruendosi dopo un certo numero di mosse. Provate l'aliante a cinque organismi (fig. 4).
Dopo la diffusione del gioco sono state fatte dalle scoperte interessantissime di cui cito le più strane e affascinanti: un cannone che spara alianti (un esempio di configurazione che cresce all'infinito); alianti di lunghezze che vanno da cinque organismi fino all'infinito, ma che per viaggiare senza morire devono essere scortate da altri alianti! (ne ho vista una di cento organismi scortata da 33 alianti standard); strane macchine mangia alianti (provate a costruire un cannone che spara alianti i quali vengono mangiati da una macchina); una configurazione di 13 alianti che collidono formando un cannone che a sua volta spara alianti!!! Il mio consiglio a questo punto è di provare le figure che ho proposto e tutte quelle che vi vengono in mente, comunicandoci le vostre scoperte. Vedrete come la popolazione varia continuamente, assumendo configurazioni spesso bellissime e sempre impreviste, che si arricchiscono di simmetrie inaspettate.
Leggete gli articoli di Martin Gardner sui numeri di Scientific American (edizione americana) che vanno dall'ottobre 1970 al febbraio 1971: sono molto belli (e io gli devo molto).

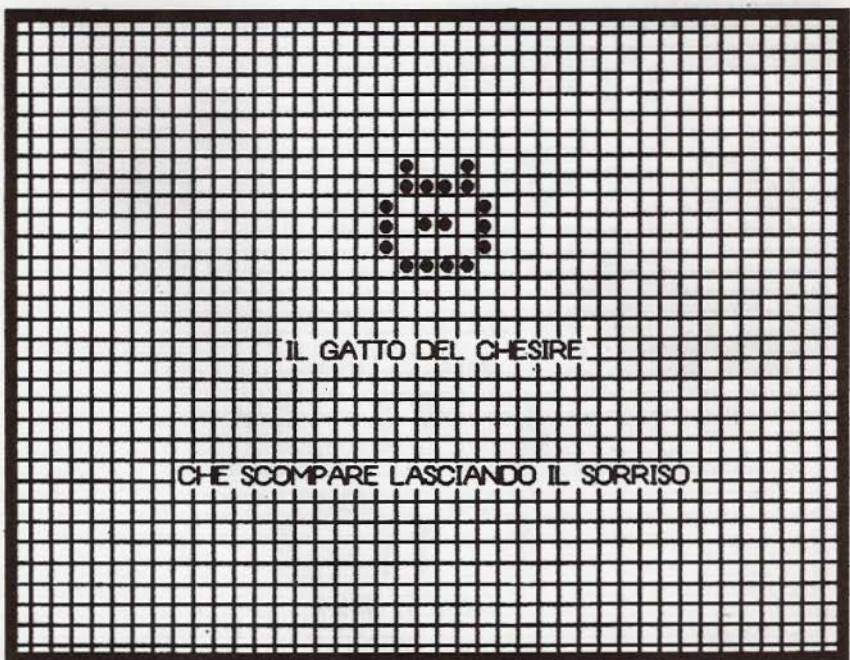
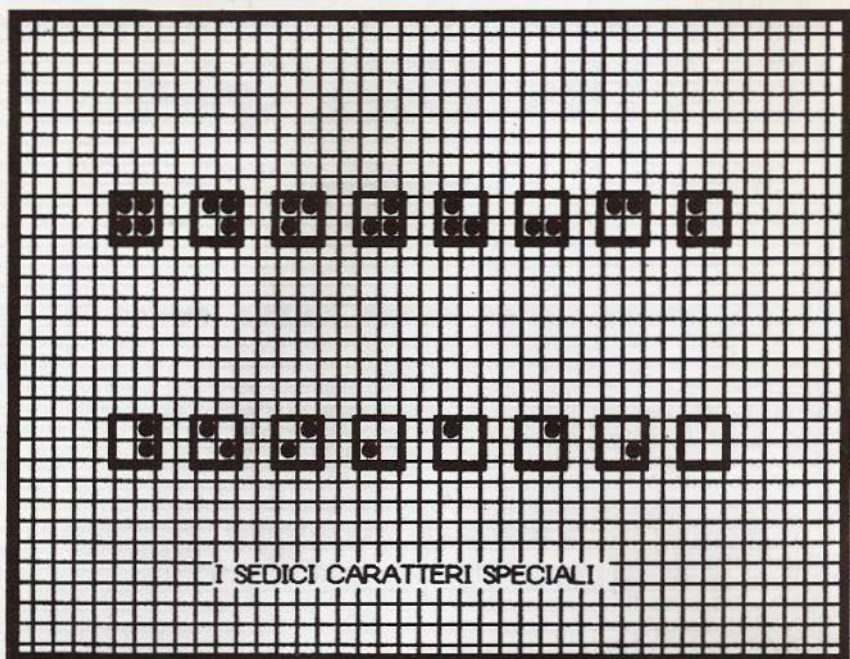
Istruzioni per l'uso

Caricate il programma e date il RUN. Dopo alcuni secondi il Programma chiederà la configurazione iniziale. Inserite le righe una alla volta, ricordando che il "Punto" significa cella vuota, e qualsiasi altro carattere significa cella piena. Se dovete inserire delle righe completamente vuote, digitate almeno un punto. Per terminare premete return a vuoto. Il Programma vi avverte in caso di estinzione totale ed in caso di sconfinamento dal bordo dello schermo.

Descrizione del programma

I due grossi problemi che si presentano nella realizzazione di LIFE riguardano il numero massimo di organismi visualizzabili sullo schermo, e la velocità di elaborazione, che evidentemente dipende da questo nūmero.
La pagina schermo del CBM 64 limita a mille, in bassa risoluzione,

ESEMPI DI CONFIGURAZIONI STABILI OD OSCILLANTI

I SEDICI CARATTERI SPECIALI

IL GATTO DEL CHESIRE
CHE SCOMPARE LASCIANDO IL SORRISO.

il numero di caratteri visualizzabili sullo schermo (40 righe * 25 colonne). Andare in alta risoluzione con il Basic residente comporterebbe l'uso di una serie di routine che rallenterebbero troppo il Programma, senza contare il grande dispendio di memoria che ne deriverebbe (provate ad immaginare una matrice 230*200). Inoltre la visualizzazione delle popolazioni sarebbe poco definita.

Il sistema da me adottato è una via di mezzo, che porta il numero di righe e colonne rispettivamente a 50 e 80, mediante l'utilizzo di una serie di caratteri ridefiniti. Ognuno di questi caratteri rappresenta una delle 16 possibili combinazioni di cerchietti di 4 Pixel di diametro, dentro un quadrato di 8 Pixel di lato (vedi figura 5). Una volta eseguiti i calcoli relativi alle nascite e alle morti, il programma, invece di stampare una cella alla volta, ne esamina 4 insieme, e tramite una codifica binaria sceglie quale dei 16 caratteri rappresenta quella combinazione di 4 celle. In questo modo il numero di organismi rappresentabili arriva a 4000 (in realtà sono di meno dato che due righe sono occupate dal testo).

Il secondo problema è di più difficile soluzione, rimanendo in ambiente Basic. Per ognuna delle 4000 celle, devono, per ogni generazione, essere contati gli organismi adiacenti. Il programma minimizza questo tempo, scegliendo ogni volta il più piccolo rettangolo in cui è significativo eseguire questo controllo. Anche la stampa avviene in un rettangolo simile. Nonostante tutto, la lentezza può in alcuni casi essere esasperante. L'unica soluzione rimane la compilazione in linguaggio macchina. Il più diffuso compilatore Basic per il CBM 64 è il PetSpeed, che è anche il migliore, ma purtroppo porta la lunghezza del programma da 10 a 105 settori! Questo crea un altro problema: il chip video può indirizzare 16K alla volta, e in questi 16K devono esserci contemporaneamente la pagina schermo e i caratteri speciali. Si è reso dunque necessario spostare tutte e due le cose nei (mitici) 4K disponibili dopo la ROM Basic.

Ma vediamo il listato:

1: abilita il tasto di STOP nel caso di una compilazione con PetSpeed;
2: va alla routine di inizializzazione caratteri e schermo;
3-30: inizializzazione variabili;
35-80: input configurazione iniziale;
85-180: riempie la matrice iniziale, centrando la configurazione sullo schermo. Ogni organismo è indicato con un uno, ogni cella vuota con uno zero;
190-220: messaggi di inizio schermata;
230-290: riconferma la matrice per la stampa, mettendo i nati (indicati con 3) a 1 e i morti (indicati con 2) a zero;
291-294: decodifica 4 celle adiacenti e stampa il carattere speciale corrispondente;
299-307: controlla se la configurazione è uscira da uno dei bordi, nel qual caso setta 19 = 1;
309-650: esegue i controlli sulle nascite e le morti ponendole rispettivamente uguali a 3 e 2;
1000-1050: spegne la routine che controlla automaticamente la tastiera; accende la ROM dei caratteri; copia i primi 64 caratteri dalla locazione 53248 alla locazione 51200; spegne la ROM dei caratteri; riattiva la routine di tastiera;
1060-1100: cambia il numero di Bank accessibile dal chip video, portandolo a 0 a 3; cambia l'indirizzo della memoria di schermo portandolo a 49152; cambia l'indirizzo della memoria dei caratteri portandolo a 51200;
1110-1112: definisce i caratteri speciali, ricopiando i DATA nelle locazioni corrispondenti;
2000-2150: codifica decimale dei nuovi caratteri. Ogni riga di DATA è un carattere.

```
 1 REM !ES
 2 GOSUB 1000
 3 DIM A(44,76),B$(44)
 5 PRINT"[CLEAR]":X2=44:Y2=76:P=0
   :G=1:I9=0
30 C=1:POKE 53280,246:POKE 53281,
   246
35 PRINT"CONFIGURAZIONE INIZIALE:
   "
40 INPUT B$(C):L=LEN(B$(C)):IF L>
   73 THEN L=73
50 IF L=0 THEN PRINT"[CLEAR]":GOT
   O 80
55 B$(C)=MID$(B$(C),1,L)
60 C=C+1
70 IF C<X2 THEN 40
80 IF C=1 THEN 5
85 L=0
 90 FOR X=1 TO  C
100 IF LEN(B$(X))>L  THEN  L=LEN(B
    $(X))
110 NEXTX
120 X1=22-C/2
130 Y1=38-L/2
140 FOR X=1 TO C-1
150 FOR Y=1 TO  LEN(B$(X))
160 IF MID$(B$(X),Y,1)<>"." THEN A
    (X+X1,Y+Y1)=1:P=P+1
170 NEXTY
180 NEXTX
190 X2=X1+C
200 IF P<>0 THEN 210
205 PRINT"[CLEAR]GENERAZIONE:";G;"
      POPOLAZIONE ESTINTA"
208 FOR X=1 TO 1000:NEXTX: RUN
210 PRINT"[HOME]GENERAZIONE:";G;"
```

```
    POPOLAZIONE:";RIGHT$("0000"+ST
    R$(P),4);
215 IF I9<>0 THEN PRINT" [RVS]BORD
    I"
216 PRINT
218 X3=44:Y3=76:X4=1:Y4=1
220 G=G+1
230 FOR X=X1 TO X2
250 FOR Y=Y1 TO Y2
253 IF A(X,Y)=2 THEN A(X,Y)=0
256 IF A(X,Y)=3 THEN A(X,Y)=1
260 IF A(X,Y)<>1 THEN 270
262 IF X<X3 THEN X3=X
264 IF X>X4 THEN X4=X
266 IF Y<Y3 THEN Y3=Y
268 IF Y>Y4 THEN Y4=Y
270 NEXTY
290 NEXTX
291 FOR X=1 TO X2 STEP 2:PRINT
292 FOR Y=1 TO Y2 STEP 2:MU=A(X,Y)
    *8+A(X,Y+1)*4+A(X+1,Y)*2+A(X+1
    ,Y+1)
293 PRINTA$(MU);
294 NEXTY:NEXTX
299 X1=X3:X2=X4:Y1=Y3:Y2=Y4:I9=0
301 IF X1<3 THEN X1=3:I9=-1
303 IF X2>42 THEN X2=42:I9=-1
305 IF Y1<3 THEN Y1=3:I9=-1
307 IF Y2>74 THEN Y2=74:I9=-1
309 P=0
500 FOR X=X1-1 TO X2+1
510 FOR Y=Y1-1 TO Y2+1
520 C=0
530 FOR I=X-1 TO X+1
540 FOR J=Y-1 TO Y+1
550 IF A(I,J)=1 THEN C=C+1
555 IF A(I,J)=2 THEN C=C+1
560 NEXTJ
570 NEXTI
580 IF A(X,Y)=0 THEN 610
582 P=P+1
584 IF C=3 THEN 620
586 IF C=4 THEN 620
588 P=P-1
590 A(X,Y)=2
600 GOTO 620
610 IF C=3 THEN A(X,Y)=3:P=P+1
620 NEXTY
630 NEXTX
635 X1=X1-1:Y1=Y1-1:X2=X2+1:Y2=Y2+
    1
640 GOTO 200
650 END
1000 POKE 56334,PEEK(56334) AND 254
1010 POKE 1,PEEK(1) AND 251
1020 FOR I=0 TO 511
1030 POKE 51200+I,PEEK(53248+I):NEX
     TI
1040 POKE 1,PEEK(1) OR 4
1050 POKE 56334,PEEK(56334) OR 1
1060 POKE 56578,PEEK(56578) OR 3
1070 POKE 56576,(PEEK(56576) AND 25
     2) OR 0
1080 POKE 53272,(PEEK(53272) AND 15
     ) OR 0
1090 POKE 648,192
1100 POKE 53272,(PEEK(53272) AND 24
     0) OR 2
1110 FOR I=51480 TO 51535:READ WW:P
     OKE I,WW:NEXTI
1111 FOR I=51416 TO 51471:READ WW:P
     OKE I,WW:NEXTI
1112 FOR I=51544 TO 51559:READ WW:P
     OKE I,WW:NEXTI
1115 DIM A$(16)
1120 A$(0)=" ":A$(1)="#":A$(2)="$":
     A$(3)="%"
1130 A$(4)="&":A$(5)="[":A$(6)="£":
     A$(7)="]"
1140 A$(8)="´":A$(9)="(":A$(10)=")"
     :A$(11)="↑"
1150 A$(12)="←":A$(13)="!":A$(14)="
     +":A$(15)=","
1160 RETURN
2000 DATA 0,0,0,0,6,15,15,6
2010 DATA 0,0,0,0,96,240,240,96
2020 DATA 0,0,0,0,102,255,255,102
2030 DATA 6,15,15,6,0,0,0,0
2040 DATA 96,240,240,96,0,0,0,0
2050 DATA 96,240,240,96,6,15,15,6
2060 DATA 96,240,240,96,96,240,240,
     96
2070 DATA 6,15,15,6,6,15,15,6
2080 DATA 6,15,15,6,96,240,240,96
2090 DATA 6,15,15,6,102,255,255,102
2100 DATA 96,240,240,96,102,255,255
     ,102
2110 DATA 102,255,255,102,0,0,0,0
2120 DATA 0,0,0,0,0,0,0,0
2130 DATA 102,255,255,102,6,15,15,6
2140 DATA 102,255,255,102,96,240,24
     0,96
2150 DATA 102,255,255,102,102,255,2
     55,102
```

FATTORI PRIMI M.C.D. e m.c.m.

di **Mauro Massetti**

Scomposizione di numeri in fattori primi, ricerca del massimo comun divisore e del minimo comun multiplo fra due numeri

L'operazione di scomposizione di un numero nei suoi fattori primi non presenta particolari difficoltà quando si deve affrontare questo problema con l'ausilio di un calcolatore elettronico.

A tal proposito esistono diversi programmi che, sfruttando vari algoritmi, pervengono al risultato voluto.

Quello che propongo è forse il più semplice: considera il numero e lo divide per un fattore che, partendo dal valore 2, viene costantemente incrementato di 1 sino a che il resto della divisione non risulti uguale a zero.

A questo punto il numero da considerare è il risultato della divisione e tutto ricomincia di nuovo.

Si ottiene così una scomposizione dove i vari componenti compaiono in ordine crescente di grandezza.

È di ovvia constatazione che il numero considerato, nel caso in cui non risulti divisibile esattamente per nessun fattore minore o uguale alla metà del numero stesso, sia un numero primo.

Si può affrontare quindi il problema della ricerca del massimo comun divisore (MCD) e del minimo comun multiplo (mcm) fra due numeri.

Per cominciare carichiamo i fattori di scomposizione di due numeri (trovati con il metodo sopra citato) in due vettori.

M.C.D. — m.c.m. Tutti dovrebbero sapere che per trovare il massimo comun divisore si deve considerare solamente i fattori che compaiono in entrambi i vettori con esponente minimo (si noti che tutti i fattori nei vettori hanno esponente 1!) mentre per il minimo comun multiplo vanno considerati tutti i fattori diversi fra loro più quelli uguali con esponente massimo.

Pare quindi che la ricerca del minimo comun multiplo possa presentare problemi più seri che non quella per il massimo comun divisore ma, utilizzando il metodo proposto si può facilmente constatare che, se si ricerca prima il massimo comun divisore, con gli elementi a nostra disposizione la ricerca del minimo comun multiplo diviene immediata.

Per esempio cerchiamo il massimo comun divisore di due numeri qualsiasi:

100 e 90.

La scomposizione in fattori primi fornisce:

Tabella 1

Numero	Fattore
100	2
50	2
25	5
5	5

$$\vec{A} = \begin{vmatrix} 2 \\ 2 \\ 5 \\ 5 \end{vmatrix}$$

e

Numero	Fattore
90	2
45	3
15	3
5	5

$$\vec{B} = \begin{vmatrix} 2 \\ 3 \\ 3 \\ 5 \end{vmatrix}$$

Poniamo ora il valore del MCD uguale a 1 ed operiamo come segue: consideriamo il primo fattore in

COMPUTER QUESTO MESE È QUESTO

Computer

N.69 - lire 3000 il "newsmagazine" dell'informatica

Tutte le stampanti del mercato

Cosa vedere al prossimo Smau

Data base: come sceglierli

Ed ecco il transputer

Benchmark: il People Olympia un anno dopo

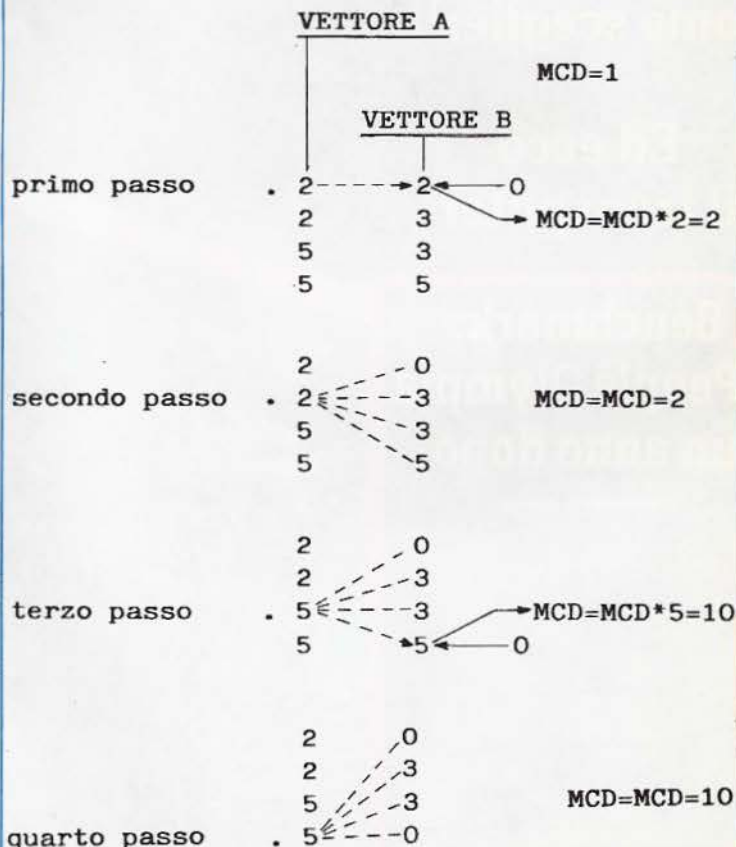
A (nel nostro esempio = 2) e confrontiamolo con tutti i valori in B sino a che non ne troviamo uno uguale (per caso forse strano è il primo); se questa condizione si verifica moltiplichiamo il MCD per il fattore B (j) (per cui si farà 1*2) e poniamo zero in B (j) per evitare che nella scansione successiva venga ripreso in considerazione lo stesso valore.

Ripetiamo quanto sopra per ogni fattore in A ed alla fine avremo il MCD cercato.

In pratica il programma procede così:

A questo punto ci si accorge che, grazie alla precedente metodologia operativa, il prodotto dei fattori diversi da zero rimasti nei vettori A e B è il mcm cercato.

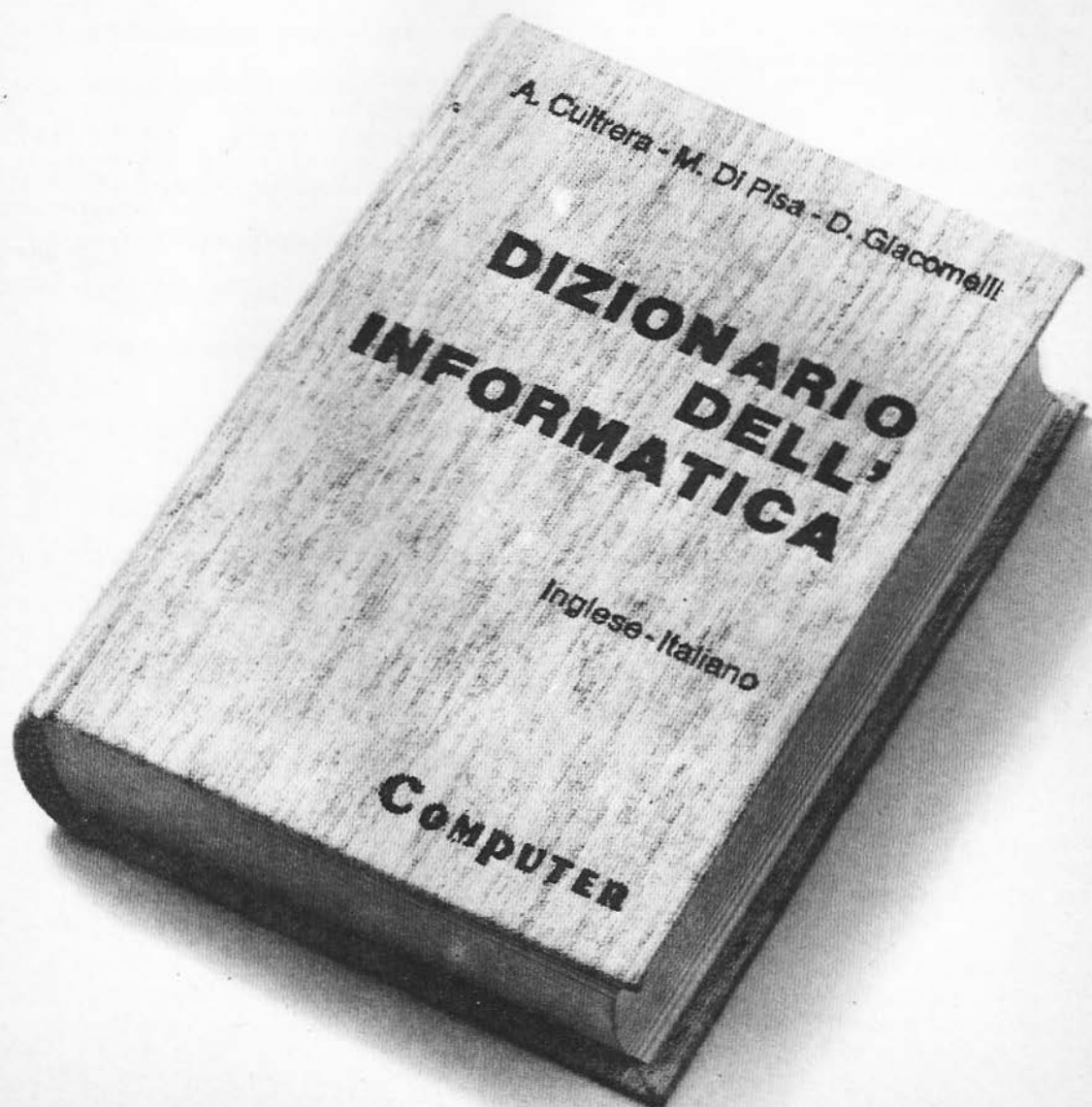
È sufficiente quindi scandire gli elementi di entrambi i vettori e moltiplicare fra loro quelli significativi per risolvere anche l'ultima proposizione.

A conclusione alleghiamo il listato del programma che volutamente è stato corredato di numerose REM per facilitare la comprensione delle funzioni delle varie routines.

Computer questo mese é questo...

E QUESTO ....

```
100 REM *************************
    *****
105 REM *
    *
110 REM * SCOMPOSIZIONE DI UN NUM
    ERO *
115 REM * IN FATTORI PRIMI
    *
120 REM * CALCOLO DEL MD E DEL MM
    *
125 REM * FRA DUE NUMERI
    *
130 REM *
    *
135 REM * AUTORE: MAURO MASSETTI
    *
140 REM *
    *
145 REM *************************
    *****
150 REM
155 REM  **    INIZIO PROGRAMMA
     ***
160 REM
165 DIM  A(50),B(50)
170 REM
175 REM  *********************
180 REM  * INPUT PRIMO NUMERO *
185 REM  *********************
190 INPUT "[CLEAR]IMPOSTARE IL PR
    IMO NUMERO        [7 LEFT]";N:
    SW=0
195 FLAG=0
200 KO=1
205 PRINT
210 A(0)=N
215 I=2
220 REM  ***********************
    *****
225 REM  * FAT. SCOMPOSIZIONE MIN
    IMO *
230 REM  ***********************
    *****
235 REM  * RICERCA DEI FATTORI PR
    IMI *
240 REM  ***********************
    *****
245 IF I=N THEN 350
250 IF I>N THEN 190
255 IF ((N/I)-INT(N/I))=0 THEN 27
    5
260 IF I>N/2 THEN I=N:GOTO 350
265 I=I+1
270 GOTO 245
275 FLAG=1
280 X=N
285 GOSUB 780
290 PRINTCHR$(98);I
295 REM  ***********************
    ***
300 REM  * CAR. FATTORE SU VETTOR
    E *
305 REM  ***********************
    ***
310 N=N/I
315 A(KO)=I
320 KO=KO+1
325 IF N=1 THEN 360
330 GOTO 215
335 REM  ***********************
    *****
340 REM  * SEGNALAZIONE NUMERO PR
    IMO *
345 REM  ***********************
    *****
350 IF FLAG=0 THEN PRINTN;"E' N.P
    RIMO":A(1)=N:KO=2:GOTO 360
355 GOTO 275
360 PRINT
365 PRINT
370 TN=KO-1
375 REM  ***********************
380 REM  * INPUT SECONDO NUMERO *
385 REM  ***********************
390 INPUT "[HOME]IMPOSTARE IL SEC
    ONDO NUMERO        [7 LEFT]";M
    :SW=1
395 FLAG=0
400 KO=1
405 PRINT
410 B(0)=M
415 I=2
420 REM  ***********************
    *****
425 REM  * FAT. SCOMPOSIZIONE MIN
    IMO *
430 REM  ***********************
    *****
435 REM  * RICERCA DEI FATTORI PR
    IMI *
```

```
440 REM  ************************
    *****
445 IF I=M THEN 550
450 IF I>M THEN 390
455 IF ((M/I)-INT(M/I))=0 THEN 47
    5
460 IF I>M/2 THEN I=M:GOTO 550
465 I=I+1
470 GOTO 445
475 FLAG=1
480 X=M
485 GOSUB 780
490 PRINTCHR$(98);I
495 REM  ************************
    *****
500 REM  * CARIC. FATTORE SU VETT
    ORE *
505 REM  ************************
    *****
510 M=M/I
515 B(KO)=I
520 KO=KO+1
525 IF M=1 THEN 575
530 GOTO 415
535 REM  ************************
    *****
540 REM  * SEGNALAZIONE NUMERO PR
    IMO *
545 REM  ************************
    *****
550 IF FLAG=0 THEN PRINT TAB(20*S
    W)M"E' N.PRIMO":B(1)=M:KO=2:G
    OTO 575
555 GOTO 475
560 REM  ************************
    ***
565 REM  * RICERCA DEL MD E DEL M
    M *
570 REM  ************************
    ***
575 PRINT
580 PRINT
585 TM=KO-1
590 MD=1
595 MM=1
600 REM  ************************
    ****
605 REM  * SCANSIONE VETTORI PER
    MD *
610 REM  ************************
    ****
615 FOR I=1 TO TN
620 FOR J=1 TO TM
625 REM  ************************
630 REM  * AGGIORNAMENTO DEL MD *
635 REM  ************************
640 IF A(I)=B(J) THEN MD=MD*B(J):
    B(J)=0:J=TM
645 NEXTJ
650 NEXTI
655 REM  ************************
    **
660 REM  * VISUALIZZAZIONE DEL MD
     *
665 REM  ************************
    **
670 PRINT"[HOME][16 DOWN]IL MASSI
    MO COMUN DIVISORE"
675 PRINT"FRA I NUMERI :"
680 PRINTSTR$(A(0));" E ";STR$(B(
    0));"  E': [RVS] ";STR$(MD);"
     [RVOFF]"
685 REM  ************************
    *****
690 REM  * SCANSIONE VETTORI PER
    MM *
695 REM  ************************
    ****
700 FOR I=1 TO TN
705 IF A(I)<>0 THEN MM=MM*A(I)
710 NEXTI
715 FOR I=1 TO TM
720 IF B(I)<>0 THEN MM=MM*B(I)
725 NEXTI
730 REM  ************************
    **
735 REM  * VISUALIZZAZIONE DEL MM
     *
740 REM  ************************
    **
745 PRINT"IL MINIMO COMUN MULTIPL
    O"
750 PRINT"FRA I NUMERI :"
755 PRINTSTR$(A(0));" E ";STR$(B(
    0));"  E': [RVS] ";STR$(MM);"
     [RVOFF]"
760 PRINT
765 INPUT "CONTINUI(S/N)";S$
770 IF LEFT$(S$,1)="S"  THEN 190
775 END
780 X$=STR$(X)+" "
785 X=11-LEN(X$)
790 PRINT TAB(20*SW)SPC(X);X$;:RE
    TURN
```

Da AAEa ZOOMING 7065 VOCI NEL NUOVO DIZIONARIO DELL'INFORMATICA

UN DIZIONARIO COSI' SOLO COMPUTER POTEVA FARLO

Made in Italy

Il primo grande dizionario d'informatica non tradotto dall'inglese, del quale un grande editore americano come Auerbach abbia richiesto i diritti d'autore per il mercato statunitense, finalmente è disponibile. Sulla rivista la pubblicazione delle dispense proseguirà fino alla Z, ma chi vuole disporre fin d'ora dell'opera completa, rilegata in cartone telato, può richiederla alla redazione e profittare dell'offerta di lancio.

Migliaia di voci in inglese ed in italiano con le definizioni dei principali organismi di standardizzazione internazionali (IEEE, ISO, ANSI, ecc.) fanno di questo volume un indispensabile strumento di lavoro per quanti operano professionalmente nell'edp.

Anni di lavoro dei tre coautori e dell'intera redazione di "Computer" testimoniamo un impegno che nessun altro editore italiano, grande medio o piccolo, aveva mai tentato.

Aggiornamento telefonico personalizzato

E, come se non bastasse, la redazione di Computer s'impegna a rispondere singolarmente a tutte le richieste degli acquirenti del dizionario per tutte le voci nuove che non dovessero figurare in questa prima edizione ed a pubblicarne un aggiornamento periodico.

Spettabile redazione,
desidero ricevere il "Dizionario dell'informatica" da voi edito e profittare della speciale offerta di lancio

☐ *Una copia del dizionario a lire 84.000*
☐ *Una copia del dizionario + l'abbonamento a Computer per un anno, a partire dalla data dell'ordine, al prezzo complessivo di lire 94.000*
☐ *Una copia del dizionario + l'abbonamento per un anno a Computer ed a Computer Club per complessive lire 100.000*
☐ *Ho già rinnovato l'abbonamento a Computer, pertanto dedurrete l'importo da me pagato dal prezzo cumulativo dizionario + abbonamento di lire 94.000.*

☐ A tale scopo invio in allegato assegno n° .... Banca .... per l'importo di lire ....
☐ Ho versato oggi stesso l'importo di lire .... sul vostro c/c postale n. 11909207
☐ Pagherò a ricevimento della fattura, che vi prego intestare come segue:
Ditta/o ragione sociale: ....
Via .... n° .... CAP ....
Città .... Partita IVA ....

Il mio indirizzo è il seguente: Nome .... Azienda ....
Qualifica ....
Via .... N° .... CAP .... Città ....
Tel. .... F Firma ....

Vendo numerosissimi **programmi** per Commodore 64, giochi, utility oltre 200 titoli. Scrivere a Paolo Lambri - Via Alfieri 60 - Sesto San Giovanni (Milano).

Vendo / Compro Consolle Atari 2600 con josticks e paddles più 8 cassette (tra cui Decathlon - Kangaroo e altri) ancora imballato ad un prezzo d'occasione. Compro Commodore 64 o computer SEGA SC 3.000 alle condizioni sopra citate. (Barzon Furio - Via per Nogarè n. 21 - 32100 Belluno - Tel. 0437 / 26681 ore pasti).

Vendo / Compro / Cambio. Cambio molti programmi per Commodore 64 richiedere il listato, prezzi da regalo. Per acquisti superiore alle 50.000 L. 5.000 per programma (videogames) e 15.000 per l'utility (con manuale a parte). Compro Floppy 1541 occasione. (Rizzi Fabrizio - Via Castello n. 3060B - 30122 Venezia - Tel. 041 / 22883 telefonare al mattino dopo le 10).

Vendo per VIC 20 2 cartucce: Adventureland, Snake byte, prezzo da concordare. (Ferrari Fabio - Via Araldi n. 5 - 29100 Piacenza - Tel. 0523 / 753283. Scrivere o telefonare dopo le ore 19).

Vendo numerosi programmi per Commodore 64. Sono interessato anche a cambi e acquisti di altri programmi. Telefonare o scrivere inviando eventuali liste a: Penati Ferrerio Ottavio - Via L.L. Cadorna, 12 - 22053 Lecco - Tel. 0341 / 368360 (mattino e ore pasti).

Vendo / Cambio software per CBM 64, circa 200 programmi e Commodore VIC 20 circa 1.000 programmi per contatti scrivere o telefonare (Zanella Lionello - Via Virgilio 21 - 74025 Marina di Ginosa (Taranto) Tel. 099 / 677090 ore pasti).

Vendo / Cambio per Commodore 64 giochi e utility su disco o nastro. Annuncio sempre valido. Scrivere o telefonare (Parissi Eraldo - Viale dei Mughetti n. 36 - 10151 Torino - Tel. 011 / 734354 ore pasti).

Vendo pocket group lede per VIC 20, Cabinet alimentato d'espansione super expander, Aid's programmer. Sharp PC 1500 + stampante CE 150. Per CBM 64 centinaia di programmi di ogni tipo a prezzi modesti. Hewlett Packard HP 150 ottimi programmi originali. (Pocket Group Club utenti Commodore Puglia - Via Amoruso n. 34 - 70124 Bari).

Vendo / Cambio programmi per C64 (tutti in LM). Telefonate o chiedete la lista. (Mauro Di Cecca - Via De Rubeis n. 14 - 33043 Cividale - Udine - Tel. 0432 / 730912).

Vendo giochi per CBM 64: pole position, puc man, hunchback, mondial soccer, congo bongo, falcon patrol su una cassetta a L. 20.000 (spese di spedizione comprese). Mandatemi i soldi, spedirò la cassetta con tutti i giochi. (Stefano Caracciolo - C.so Italia n. 281E - 16145 Genova - Tel. 010 / 300453 ore 12-14).

Vendo per CBM centinaia di programmi in LM a prezzi incredibili sia su nastro che su disco. Es.: Zaxxon L. 5.000, Blue Max L. 3.000, Musicconstructionset L. 12.000, Simon Basic L. 2.500. (Fugazzola Davide - Via B. D'Alviano 2 - 20146 Milano - Tel. 02 / 4229466 ore pasti).

Vendo per Commodore 64 Simon's Basic + 3 dimostrazioni L. 60.000. Vendo anche cassetta con Turb tape (che rende il caricamento 10 volte più veloce) + 40 stupendi giochi (tutti con turbo) a L. 200.000, per registratore 1530 model C2N (Lucio Fiorentino - Via Gramsci 5 - 80122 Napoli - Tel. 081 / 680150 ore 20-22).

Vendo / Cambio tutti i programmi per Commodore 64. Posseggo i migliori giochi a prezzi stracciati. Vendo inoltre Simon's Basic su cassetta a L. 10.000 e tanti altri spedico lista gratis. (Galle Domenico - Via Cida Guido n. 36 - 88029 Serra San Bruno - Tel. 0963 / 71210).

Vendo Corso Basic per CBM 64, originale Commodore, interamente in italiano, comprendente manuale + 2 cassette, il tutto a L. 40.000 (Ivano Ursini - Viale J.F. Kennedy n. 96 - 65100 Pescara - Tel. 085 / 76049 ore 13,30-17).

Vendo / Compro / Cambio vastissima gamma di programmi per Vic 20, gestionali, totocalcio e oltre 200 giochi, molte copie di cartucce. (Fernando Benini - Via E. Pazzi n. 16 - 48100 Ravenna).

Vendo / Cambio moltissimo software per il Commodore 64, mandatemi le vostre liste, invierò le mie. (Carlo Micheli - Via San Primo n. 6 - 20121 Milano - Tel. 02 / 796868 nel pomeriggio).

Vendo tutti i programmi esistenti al mondo per C64: Magic Desk, Koala Painter, con joistick. Infiniti giochi meravigliosi. Easy Script con manuale in italiano, Superbase-Multidata e tantissimi programmi per ingegneria 373, chiedere lista (Bifolchi Giordano - Via Per Pienza n. 17 - 53043 Montepulciano - Tel. 0578 / 716397 - 757907 ore 10,30-23,30).

Vendo / Cambio programmi VIC 64. (Fernando Forner - Via Valperga Caluso n. 21 - 10125 Torino - Tel. 011 / 6506538).

Vendo / Cambio / Compro programmi per Commodore 64. Prezzi stracciati. Richiedere elenco e/o inviare il proprio (Gino Uglietti - Via Strambio 108 - 27011 Belgioioso - PV).

Vendo Commodore VIC 20, 7 mesi di vita + Registratore C2N + 100 giochi su cassetta + 2 cartucce (Cosmic Grunghe e Speed Bingo Matt) + joistick + manuale - istruzioni in italiano. Ancora in garanzia, tutto a L. 320.000 (tratt.). (Stefano Mungo - Via Val Pellice n. 51 - 00141 Roma - Tel. 8124044 ore 13-15, 19-21).

Vendo / Compro programmi per VIC 20 Basic e L.M. con espansione o senza. Richiedere elenco. (Bruno Gandolfi - Via P. Calamandrei n. 1 - 14049 Nizza Monferrato - Asti - Tel. 0141 / 727216 ore 12,30-13,30, 20-22).

Vendiamo programmi su cassetta per Commodore 64 di giochi e utility vari, ne disponiamo circa 300. Si garantiscono prezzi modici. Siamo interessati inoltre ad ampliare il numero degli affiliati del nostro giovanissimo Club. Richiedere lista programmi od informazioni. (C.C.C. Commodore Club Cesena c/o Maestri Maurizio - Via P. Genocchi n. 492 - 47023 Cesena - Forlì).

Vendo / Compro / Cambio software per C64, oltre 300 programmi disponibili, giochi, utility, offerta sempre valida. Scrivere inviando liste o telefonare ore pasti. Disponibilità su disco o nastro. (Lorenzo Parolo - Via Pablo Neruda n. 4 - 27028 San Martino Sicc. - Pavia - Tel. 0382 / 25086 - 498757).

Vendo per VIC 20 i migliori programmi oggi in circolazione. Invio catalogo dettagliato (36 pagine). Cedo anche traduzioni istruzioni Superexp - Toolkit. (Luigi De Negri - Via Puggia n. 22 - 16131 Genova).

Vendo per CBM 64 cassetta contenente i seguenti giochi: Donkey Kong, Q Bert, Frogger, Centipede, Arcadia, Super Pipeline, tutti registrati con il Turbo tape 64 (compreso in cassetta) che ha la capacità di aumentare la velocità del registratore di 16 volte. Il tutto a L. 25.000 (comprese spese postali). (Salvo D'Urso - Via La Farina n. 3d - 95018 Riposto - Catania - Tel. 095 / 936075 ore 13-17).

Cambio per C64 oltre 100 programmi sia su disco che nastro. Inviate la vostra lista e vi risponderò con la mia. È gradito l'invio di un francobollo. (Francesco Punto - Via Tufanelli 18 - 80046 San Giorgio a Cremano (Napoli) - Tel. 081 / 7716827 ore 21).

Compro Commodore 64, acquisto o scambio programmi su cassetta soprattutto ingegneria civile, gestionali, utility. Inviare lista con caratteristiche eprezzi. (Massimiliano Carabelli - Via Australia n. 15 - 00144 Roma - Tel. 06 / 5924220 ore pasti).

Compro cassette per VIC 20 pressapoco al modico prezzo di L. 10.000 ogni cinque. Vorrei la lista. (Giuseppe Rosatella - Via Regina Margherita n. 8 - 03012 Anagni - Tel. 727137 ore 10,30).

Compro manuale d'istruzione di Screen Graphic e Sam Reciter per CBM 64. Riviste Personal Software (da 1 a 19) e Commodore Computer Club (da 1 a 4). (Coffano Giuseppe - Via Orazio Flacco n. 5 - 10024 Moncalieri - Torino - Tel. 011 / 645923).

Compro Commodore C64 senza registratore + floppy disk in buono stato al miglior offerente. (anonimo di Milano - Tel. 02 / 6438566 nelle ore serali).

Compro programmi didattici d'utilità per C64. Sono interessato inoltre a scambio idee, esperienza e programmi. (Ivol Giovanni - Via Martiri della Libertà n. 34 - 25018 Montichiari (Brescia) - Tel. 030 / 964444 ore serali).

Compro cassetta con programmi per il Totocalcio (Commodore 64). Tale programma deve prevedere limitazioni minime e massime per l'uscita dei segni 1,2 e X, limitazione minime e massime per la consecutività dei segni 1,2 e X rispetto ad un dato pronostico. Dispositivo che consenta di valutare il punteggio conseguito una volta che si inserisca nel computer la colonna vincente. (Antonio Zanghi - Via Carnazza n. 53 - 95125 Catania).

Se vuoi abbonarti

Registrate il mio abbonamento annuale a Commodore.

☐ Ho versato oggi stesso il canone di Lire 25.000 a mezzo c/c postale nº 31532203 intestato a: Commodore Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

☐ Accludo assegno per lire 25.000 banca .... nº .... a favore di Commodore Systems Editoriale

Il mio computer è: VIC 20 ☐, C 64 ☐, altro (specificare) ....

Ho ☐ / non ho ☐ la stampante, ma voglio ☐ comprarla.

Preferisco programmi di gioco ☐, didattici ☐, d'utilità ☐, altro ....

Nome .... Cognome ....

Via .... nº .... CAP. ☐☐☐☐☐ Città ....

Tel. ....

Se vuoi collaborare

Registratemi fra i collaboratori regolari di Commodore.

A titolo di prova vi invio un articolo e la cassetta col programma "........ ...." di cui vi garantisco l'assoluta originalità autorizzandovene la pubblicazione.

☐ Scrivetemi all'indirizzo sottoindicato ....

Nome ....

Via .... Nº ....

Tel. .... CAP .... Città ....

Se vuoi un consiglio o consigliarci

HELP ....

Nome ....

Via .... nº .... CAP. ☐☐☐☐☐ Città ....

Tel. .... Orario ....

Il mio computer è configurato:

			Sono in possesso	No	Ho intenzione di acquistare
Vic 20 ☐	espanso a ..... K		☐	☐	☐
C 64 ☐			☐	☐	☐
Floppy ☐	quale: 1541 ☐	altro	☐	☐	☐
Stampante ☐	quale: MPS801 ☐	altro	☐	☐	☐
Plotter ☐	quale: 1520 ☐	altro	☐	☐	☐
Registratore ☐	quale: 1530 ☐	altro	☐	☐	☐

Televisore ☐, TV-Monitor ☐, Monitor ☐, Colore ☐, B/N ☐

Nome .... Cognome ....

Via .... nº .... CAP. ☐☐☐☐☐ Città ....

Tel. ....

Se vuoi vendere o comprare

Vendo ☐ Compro ☐ ....

Nome ....

Via .... nº .... CAP. ☐☐☐☐☐ Città ....

Tel. .... Orario ....

Da inviare in busta chiusa a:

Spett.le rivista
Commodore
Systems Editoriale

v.le Famagosta, 75
20142 Milano

Si, voglio abbonarmi

Da inviare in busta chiusa a:

Spett.le rivista
Commodore
Systems Editoriale

v.le Famagosta, 75
20142 Milano

Si, voglio collaborare

Da inviare in busta chiusa a:

Spett.le rivista
Commodore
Systems Editoriale

v.le Famagosta, 75
20142 Milano

Si, chiedo consiglio

Da inviare in busta chiusa a:

Spett.le rivista
Commodore
Systems Editoriale

v.le Famagosta, 75
20142 Milano

Si, voglio votare

Da inviare in busta chiusa a:

Spett.le rivista
Commodore
Systems Editoriale

v.le Famagosta, 75
20142 Milano

Si vendo/ compro